

Choisir le bon co-équipier pour la génération coopérative de texte

Antoine Chaffin, Thomas Scialom, Sylvain Lamprier, Jacopo Staiano,
Benjamin Piwowarski, Ewa Kijak, Vincent Claveau

▶ **To cite this version:**

Antoine Chaffin, Thomas Scialom, Sylvain Lamprier, Jacopo Staiano, Benjamin Piwowarski, et al.. Choisir le bon co-équipier pour la génération coopérative de texte. TALN 2022 - 29e conférence sur le Traitement Automatique des Langues Naturelles, Jun 2022, Avignon, France. pp.12-26. hal-03701506

HAL Id: hal-03701506
https://hal.science/hal-03701506

Submitted on 24 Jun 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Choisir le bon co-équipier pour la génération coopérative de texte

Antoine Chaffin[1, 2] Thomas Scialom[3, 4] Sylvain Lamprier[3] Jacopo Staino[4]
Benjamin Piwowarski[3] Ewa Kijak[1] Vincent Claveau[1]
(1) Univ. Rennes, CNRS, IRISA, Rennes, France
(2) IMATAG, Rennes, France
(3) ISIR Sorbonne Université, Paris, France
(4) ReciTAL, Paris, France
prenom.nom@irisa.fr, prenom.nom@isir.upmc.fr, prenom@recital.ai

RÉSUMÉ

Les modèles de langue génèrent des textes en prédisant successivement des distributions de probabilité pour les prochains tokens en fonction des tokens précédents. Pour générer des textes avec des propriétés souhaitées (par ex. être plus naturels, non toxiques ou avoir un style d'écriture spécifique), une solution — le décodage coopératif — consiste à utiliser un classifieur lors de la génération pour guider l'échantillonnage de la distribution du modèle de langue vers des textes ayant la propriété attendue. Dans cet article, nous examinons trois familles de discriminateurs (basés sur des *transformers*) pour cette tâche de décodage coopératif : les discriminateurs bidirectionnels, unidirectionnels (de gauche à droite) et génératifs. Nous évaluons leurs avantages et inconvénients, en explorant leur précision respective sur des tâches de classification, ainsi que leur impact sur la génération coopérative et leur coût de calcul, dans le cadre d'une stratégie de décodage état de l'art, basée sur une recherche arborescente de Monte-Carlo (MCTS). Nous fournissons également l'implémentation (*batchée*) utilisée pour nos expériences.

ABSTRACT

Choosing The Right Teammate For Cooperative Text Generation

Language models (LM) generate texts by successively predicting probability distributions for next tokens given past ones. In order to generate texts with some desired properties (eg. being more natural, non toxic, or having a specific writing style...), recent approaches use a classifier to guide the decoding of the LM distribution towards relevant texts with the expected property. In this paper, we examine three families of (transformer-based) discriminators for this task of cooperative decoding : bidirectional, left-to-right and generative ones. We evaluate the pros and cons of these different types of discriminators for cooperative generation, exploring their respective accuracy on classification tasks, their impact on the resulting sample quality and their computational performance. We also provide the batched implementation of the powerful cooperative decoding strategy used for our experiments, the Monte Carlo Tree Search, working with each discriminator for Natural Language Generation.

MOTS-CLÉS : Génération de texte, génération coopérative, décodage, Monte Carlo Tree Search.

KEYWORDS: Text generation, collaborative generation, decoding, Monte Carlo Tree Search.

1 Introduction

Les architectures à base de *transformers* (Vaswani *et al.*, 2017) permettent aux modèles de langue (LM pour *Language Models*) actuels de générer des textes très plausibles dans certaines situations (pour une discussion approfondie de ces capacités, voir (Clark *et al.*, 2021)). Pendant le décodage, étant donnée une séquence initiale de tokens (l'amorce ou *prompt*), le LM calcule une distribution de probabilité sur le vocabulaire du prochain token. Un token en est ensuite échantillonné et ajouté à la séquence initiale pour générer le token suivant, etc. Le choix du token suivant selon la distribution est généralement effectué par recherche gloutonne (*greedy search*), recherche par faisceau (*beam search*) (Dept., 2018) ou par échantillonnage top-k/p (*top p/k sampling*) (Fan *et al.*, 2018; Holtzman *et al.*, 2020) ; ces méthodes sélectionnent le token suivant uniquement selon la vraisemblance (selon le LM) de la séquence résultante, ce qui n'offre qu'un contrôle limité sur le texte finalement généré à la fin du processus de décodage.

Cependant, des discriminateurs (des classifieurs) peuvent être entraînés pour identifier une propriété spécifique d'un texte et donc fournir des informations utiles pour guider le modèle de langue vers une propriété désirée, soit lors de l'apprentissage, soit lors du décodage. Plusieurs études utilisent ainsi des discriminateurs pour distinguer les contenus réels des contenus générés (Yu *et al.*, 2017), visant par la même occasion à améliorer le réalisme des textes générés (Scialom *et al.*, 2020). D'autres cherchent à modéliser des propriétés sémantiques, par exemple la polarité du texte pour générer sur demande des textes de polarité positive ou négative (Chaffin *et al.*, 2021; Krause *et al.*, 2021; Dathathri *et al.*, 2020).

En pratique, l'utilisation du discriminateur peut être faite lors de l'apprentissage du LM, à la manière des Generative Adversarial Networks (Goodfellow *et al.*, 2020). Cependant, pour pallier les limites de ces approches adversariales sur le texte (Caccia *et al.*, 2020), certains travaux utilisent plutôt les discriminateurs pour guider le décodage au moment de la génération de manière coopérative (Holtzman *et al.*, 2018; Scialom *et al.*, 2020; Gabriel *et al.*, 2021; Bakhtin *et al.*, 2021). C'est dans ce cadre dit de décodage coopératif que ce situe notre travail.

Actuellement, les discriminateurs les plus performants reposent sur des architectures à base de transformers avec attention bidirectionnelle (Devlin *et al.*, 2019). Mais cela ne convient pas à la nature itérative du processus de génération qui nécessite de recalculer chaque état caché de la séquence entière pour chaque token supplémentaire, empêchant l'utilisation des états cachés de l'itération précédente, et entraînant un coût quadratique fonction de la longueur de la séquence. Au contraire, les transformers unidirectionnels, qui utilisent des masques de gauche à droite pour ne dépendre que des tokens passés pour l'encodage/décodage du texte (Radford *et al.*, 2019), permettent la ré-utilisation des états cachés pour les étapes suivantes, impliquant donc une complexité de calcul linéaire. Ces deux types de discriminateurs n'évaluent cependant qu'une seule séquence à la fois, donnée en entrée du modèle. Cela limite le nombre de tokens qu'il est possible de tester à chaque étape de décodage, sous peine de coûts de calcul prohibitifs. Pour résoudre ce problème, les discriminateurs génératifs (GeDi) (Krause *et al.*, 2021) récemment introduits donnent un score pour tous les tokens du vocabulaire en une seule fois, ce qui réduit considérablement le coût de l'exploration en largeur (tous les tokens peuvent être testés au temps t).

Dans cet article, nous étudions les avantages et inconvénients de ces trois types de discriminateurs pour le décodage coopératif : les discriminateurs avec attention bidirectionnelle (Devlin *et al.*, 2019), ceux avec attention unidirectionnelle (Radford *et al.*, 2019), et les discriminateurs génératifs (Krause *et al.*, 2021). Pour les évaluer dans ce cadre de génération coopérative, nous utilisons une approche

de décodage état-de-l'art inspiré du Monte Carlo Tree Search (MCTS) (Coulom, 2006). Le MCTS permet en effet de proposer une startégie de décodage non-myope, obtenant des résultats état-de-l'art dans différents cadres applicatifs (Scialom *et al.*, 2021; Chaffin *et al.*, 2021; Leblond *et al.*, 2021; Lamprier *et al.*, 2022). Nous en rappelons brièvement le principe ci-après. Nous fournissons enfin une implémentation du MCTS capable de générations par lot (*batch*) pour chaque type de discriminateur, basée sur la bibliothèque Transformers d'HuggingFace (Wolf *et al.*, 2020).

2 Contexte et motivations

2.1 Décodage coopératif et MCTS

Contrôler finement les propriétés de textes générés est utile dans de nombreux contextes. Par exemple, les LM de grande taille entraînés avec des données tout-venant sont connus pour produire des contenus toxiques et inappropriés (Bender *et al.*, 2021; Gehman *et al.*, 2020). Cela est particulièrement problématique pour beaucoup de tâches du TAL reposant sur la génération de texte, comme le Question-Answering (Nakano *et al.*, 2021; Lewis *et al.*, 2020), le résumé multi-documents (Pasunuru *et al.*, 2021), l'expansion ou la suggestion de requêtes (Claveau, 2021; Mustar *et al.*, 2022), les chatbots (Pallagani & Srivastava, 2021)...

Pour guider la génération, l'information provenant d'un discriminateur (i.e., un classifieur) peut être combinée à la distribution du générateur pour orienter le décodage vers la propriété désirée, définie par une classe du discriminateur (Chen *et al.*, 2020; Bakhtin *et al.*, 2021; Yuan *et al.*, 2021; Holtzman *et al.*, 2018; Scialom *et al.*, 2020). Par exemple, en s'inspirant de (He *et al.*, 2017; Ren *et al.*, 2017), Scialom *et al.* (2020) a proposé de ré-ordonner les séquences générées (par *beam search*) en fonction de leurs scores donnés par le discriminateur. Plus récemment, l'utilisation de l' algorithme du Monte Carlo Tree Search (MCTS) (Silver *et al.*, 2017; Coulom, 2006) a permis d'obtenir des résultats état de l'art pour différentes applications (Scialom *et al.*, 2021; Chaffin *et al.*, 2021; Leblond *et al.*, 2021; Lamprier *et al.*, 2022). Le lecteur intéressé peut se reporter à ces références pour une description détaillée, nous n'en donnons que les grands principes ci-dessous. Cet algorithme construit itérativement un arbre (dans notre cas, de génération) permettant des décisions à court terme (ajout d'un token à la séquence courante) qui semblent prometteuses à long terme (séquence complète).

Chaque itération est composée de trois étapes.

1. Premièrement, lors de la **sélection**, une recherche vers un nœud inexploré est conduite par un compromis entre l'exploitation de bonnes séquences partiellement générées (au sens d'un classifieur appris à discriminer des séquences textuelles partielles, selon leur appartenance ou non à la classe d'intérêt) et l'exploration de séquences prometteuses. Ce compromis est contrôlé par un paramètre $c_{puct} \in \mathbb{R}$ (des valeurs plus élevées signifient plus d'exploration).
2. Ensuite, une **expansion** est effectuée en créant les enfants du nœud sélectionné (ajout d'un token).
3. Enfin, la séquence correspondante est évaluée par le discriminateur et le score de chaque parent jusqu'à la racine de l'arbre est mis à jour en conséquence au cours d'une phase de **rétropropagation**. Ce score rétro-propagée est généralement calculé, à partir du nœud sélectionné, par simulation (ou **roll-out**), c'est-à-dire en développant la séquence jusqu'à sa fin (token de fin de séquence) et en évaluant la séquence complète résultante. Comme dans d'autres approches utilisant le MCTS, nous remplaçons ces *roll-out* coûteux dans nos

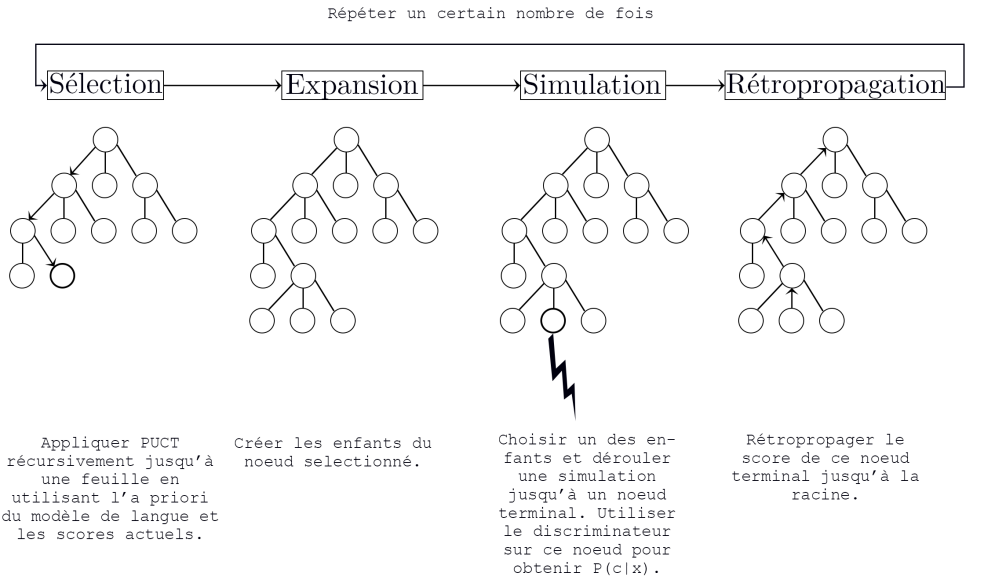


FIGURE 1 – Étapes d'une itération d'exploration par MCTS ; illustration de Chaffin *et al.* (2022).

expériences par des scores fournis par des discriminateurs entrainés sur des séquences de texte incomplètes.

On peut retrouver ces étapes dans la figure 1 illustrant l'utilisation du MCTS, avec l'étape de sélection, l'étape d'expansion, l'étape optionnelle de roll-out et l'étape de rétro-propagation du score.

Dans notre cas de génération de texte, le MCTS offre donc une stratégie d'exploration plus sophistiquée que la *beam search*, avec une vision possible à long terme (choix du token en fonction de ceux à venir ensuite). Son coût est contrôlable par le nombre d'itération, la profondeur de roll-out, et le paramètre c_{puct} contrôle le compromis exploration/exploitation.

2.2 Choix du bon co-équipier

Par défaut, les couches d'attention définies par Vaswani *et al.* (2017) sont bidirectionnelles : chaque token peut porter attention aux tokens à toutes les positions. En classification, les modèles à attention bidirectionnelle sont couramment utilisés car « *intuitivement, il est raisonnable de croire qu'un modèle bidirectionnel est strictement plus puissant que [...] un modèle gauche-droite* » (Devlin *et al.*, 2019). Cependant, la bidirectionnalité rend le modèle non auto-régressif : si un token x_t du vocabulaire $\mathcal{V}$ est ajouté à la fin d'une séquence $x_{1:t-1}$ déjà classifiée, tous les états cachés doivent être recalculés, entraînant un coût quadratique (t^2). Avec une attention unidirectionnelle (utilisée pour les LM génératifs (Radford *et al.*, 2019)), l'ajout de x_t à la suite de $x_{1:t-1}$ ne modifie pas les états cachés déjà calculés, puisque les tokens précédents n'y prêtent pas attention ; classifier $x_{1:t}$ nécessite seulement de calculer les t scores d'attention du nouveau token. Dans les tâches discriminatives courantes, cela n'a pas d'importance puisque seules des séquences entières sont manipulées. Les états cachés n'ont donc pas besoin d'être réutilisés pour une séquence suivante. Pour du décodage coopératif, où les séquences d'entrée sont la continuation de séquences déjà évaluées, l'attention unidirectionnelle laisse

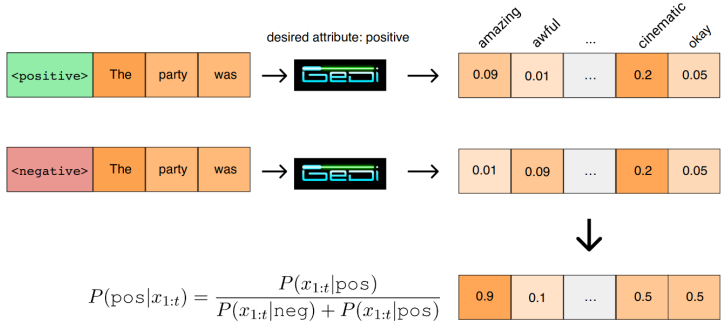


FIGURE 2 – Schéma illustrant le principe de classification de GeDi sur un problème de classification à 2 classes (*positive* vs. *negative*), tiré de (Krause *et al.*, 2021). En 2 passes, l'une avec le token de contrôle de la classe positive et l'autre avec le token de la classe négative, on obtient un score pour chacun des token du vocabulaire.

espérer des gains importants par rapport à l'attention bidirectionnelle puisqu'elle permet de réutiliser l'encodage contextuel des tokens précédents, accélérant ainsi considérablement le processus.

Cependant, même avec des discriminateurs unidirectionnels, tester toutes les suites possibles d'une séquence donnée nécessite $|\mathcal{V}|$ évaluations (*forward pass*) à chaque étape du décodage. $|\mathcal{V}|$ étant de l'ordre de la dizaine de milliers, cela est trop coûteux. Ainsi, les approches coopératives doivent contourner ce problème en limitant le nombre de séquences effectivement évaluées par le discriminateur. Par exemple, DAS (Scialom *et al.*, 2020) pré-filtre les continuations potentielles selon leur vraisemblance (donnée par le LM) (Holtzman *et al.*, 2020). Ce choix biaise nécessairement la distribution générée résultante.

Récemment, Krause *et al.* (2021) ont introduit les discriminateurs génératifs (GeDi), qui exploitent les *Class-Conditional LM* (CC-LMs) (Keskar *et al.*, 2019) pour discriminer tous les tokens de $\mathcal{V}$ à la fois. Les CC-LM conditionnent les distributions d'une séquence x à une classe c : $p(x \mid c) = \prod_t p(x_t \mid x_{1:t-1}, c)$. En supposant une distribution préalable uniforme des classes $c \in \mathcal{C}$ et $p(x_{1:T})$ étant constant pour toutes les classes, la règle de Bayes permet d'utiliser cette approche pour la discrimination : $p(c \mid x_{1:T}) \propto p(x_{1:T} \mid c)$. La figure 2 issue de l'article original de Krause *et al.* (2021) illustre ce fonctionnement. Ainsi, il suffit de $|\mathcal{C}|$ *forward pass* pour obtenir les scores de discrimination de toutes les continuations possibles de la séquence. Comme généralement $|\mathcal{C}| \ll |\mathcal{V}|$, cela rend possible la prise en compte de chaque token pour le décodage coopératif. Afin d'améliorer la capacité de discrimination de ces modèles, GeDi combine, avec un hyper-paramètre λ, la *loss* traditionnelle des modèles de langue $\mathcal{L}_g$ à une *loss* discriminative $\mathcal{L}_d$ (*cross-entropy loss* avec le LM comme discriminateur) : $\mathcal{L}_{total} = \lambda\mathcal{L}_g + (1-\lambda)\mathcal{L}_d$.

Ces trois types de discriminateurs offrent différents compromis coût/capacité, que nous étudions ici pour le décodage coopératif avec MCTS. Plus précisément, trois questions sont explorées :

1. Comment ces modèles diffèrent-ils en termes de précision de discrimination pure ?
2. Dans quelle mesure ces différences sont-elles perceptibles dans les textes générés ?
3. Comment ces méthodes se comparent-elles en termes de complexité de calcul pour le décodage coopératif ?

```
negative The Worst! A complete waste of time. Typographical
   errors, poor grammar, and a totally pathetic plot add up
   to absolutely nothing. I'm embarrassed for this author
   and very disappointed I actually paid for this book.
positive Great book This was a great book,I just could not put
   it down,and could not read it fast enough.
```

FIGURE 3 – Exemples d'entrées du jeu de données amazon_polarity, avec leur classe.

3 Étude empirique

En terme de précision, les modèles bidirectionnels devraient être supérieurs aux unidirectionnels (Devlin *et al.*, 2019; Radford *et al.*, 2019), eux-mêmes supérieurs aux générateurs discriminatifs (Yogatama *et al.*, 2017; Ng & Jordan, 2001). Pour évaluer de manière approfondie les avantages et les inconvénients de ces modèles, avec les architectures transformers les plus récentes, il est crucial que la seule différence soit la propriété étudiée (uni- vs. bidirectionnalité, discriminatif vs. génératif). Ainsi, nous utiliserons la même architecture transformer, avec une simple couche de sortie entièrement connectée au-dessus de l'*embedding* contextuel du token final, pour produire les scores de discrimination. En partant de BERT (Devlin *et al.*, 2019) comme discriminateur bidirectionnel, un masque d'attention triangulaire est appliqué pour le rendre unidirectionnel, en suivant Dong *et al.* (2019). Le discriminateur génératif (GeDi) est le même que ce dernier, la seule différence étant la taille de la couche de sortie qui passe de $(hidden_size, |\mathcal{C}|)$ à $(hidden_size, |\mathcal{V}|)$.

Les expériences sont réalisées sur deux jeux de données en anglais de Zhang *et al.* (2015) :

— amazon_polarity qui est une tâche de classification d'avis en ligne selon leur polarité (positive ou négative) ; un exemple est fourni en figure 3.

— AG_news qui est une tâche de classification thématiques d'articles avec 4 étiquettes (`world`, `sports`, `business` et `science/technology`) ; un exemple est fourni en figure 4.

Pour nos expériences, les données de chaque jeu sont divisées en deux ensembles égaux, l'un servira pour l'entraînement des modèles testés, et l'autre pour la construction des oracles servant à l'évaluation (voir section 3.2). Ces sous-ensembles sont eux-mêmes subdivisés en ensembles d'entraînement, validation et test, de taille respective de 48k, 12k, 3k8 pour AG_news, et de 50k, 15k et 5k pour amazon_polarity.

Chaque modèle est entraîné pendant 20 *epochs* en utilisant AdamW (Loshchilov & Hutter, 2019) avec les paramètres par défaut du *trainer* d'HuggingFace : *scheduler* linéaire, sans *warmup*, $\beta_1 = 0,9, \beta_2 = 0,999, \epsilon = 1e-8$. La taille des batchs est fixée au maximum tenant en mémoire d'une Quadro RTX 6000 pendant l'entraînement de GeDi : 4 pour AG_news, 8 pour amazon_polarity. L'accumulation de gradient émule une taille de batch de 128. Pour l'apprentissage de GeDi, nous avons suivi les auteurs en fixant $\lambda = 0,6$ (et nous n'avons pas observé de différence significative en fixant $\lambda = 0$ pour renforcer la capacité de discrimination).

3.1 Capacités de discrimination

Pour la génération coopérative, un discriminateur avec une bonne précision sur des séquences complètes n'est pas suffisant : une sortie informative avec des séquences non complètes est nécessaire pour que le discriminateur puisse être utilisé tout au long du processus de génération. Nous proposons

Business Veteran inventor in market float Trevor Baylis, the veteran inventor famous for creating the Freeplay clockwork radio, is planning to float his company on the stock market.

Sci/Tech Gene Blocker Turns Monkeys Into Workaholics - Study (Reuters) Reuters - Procrastinating monkeys were turned into workaholics using a gene treatment to block a key brain compound, U.S. researchers reported on Wednesday.

Sports Indians Mount Charge The Cleveland Indians pulled within one game of the AL Central lead by beating the Minnesota Twins, 7-1, Saturday night with home runs by Travis Hafner and Victor Martinez.

World Frail Pope Celebrates Mass at Lourdes LOURDES, France - A frail Pope John Paul II, breathing heavily and gasping at times, celebrated an open-air Mass on Sunday for several hundred thousand pilgrims, many in wheelchairs, at a shrine to the Virgin Mary that is associated with miraculous cures. At one point he said "help me" in Polish while struggling through his homily in French...

FIGURE 4 – Exemples d'entrées du jeu de données AG_news, avec leur classe.

donc d'étudier leurs performances en classification sur des séquences tronquées (i.e. leur précision en fonction du nombre de tokens d'entrée). Cette étude donne des informations sur la capacité du modèle à guider la génération à différents pas de temps, la principale propriété attendue pour les discriminateurs dans la génération coopérative. Notons que le discriminateur est entraîné sur des séquences de longueurs variables pour mener au mieux cette tâche.

Les résultats en figures 5 et 6 montrent que pour tous les discriminateurs, la précision augmente rapidement avec la longueur de l'entrée jusqu'à atteindre un plateau. L'ordre attendu est observé : les modèles bidirectionnels surpassent les unidirectionnels, qui surpassent les modèles génératifs. Cependant, il faut noter que l'écart est plutôt faible et n'apparaît qu'à l'approche du plateau. Privilégier les modèles bidirectionnels dans les tâches critiques en termes de précision est justifié, mais pour le décodage coopératif, il n'est pas évident que ces différences se reflètent dans la qualité des textes générés.

Notons aussi que ceci correspond à la précision sur des données *dans le domaine*. Les résultats sur des séquences aléatoires hors domaine montrent que GeDi est plus robuste aux séquences *hors domaine* : ses scores de discrimination sont beaucoup plus proches de l'incertitude maximale (i.e., $p(c|x) = 0.5$), que ceux des modèles discriminatifs qui ont tendance à favoriser une classe par rapport aux autres dans de tels cas. Cependant, cela peut ne pas avoir d'impact sur les résultats en génération, puisque de tels échantillons aléatoires ne sont pas susceptibles d'être observés pendant le décodage du MCTS, en raison de la précédence du modèle de langue à l'étape de sélection.

3.2 Qualité de génération

Nous voulons évaluer l'impact des différences vues ci-dessus sur les résultats de la génération coopérative. Nous adoptons pour cela le décodage avec MCTS tel qu'implémenté dans PPL-MCTS

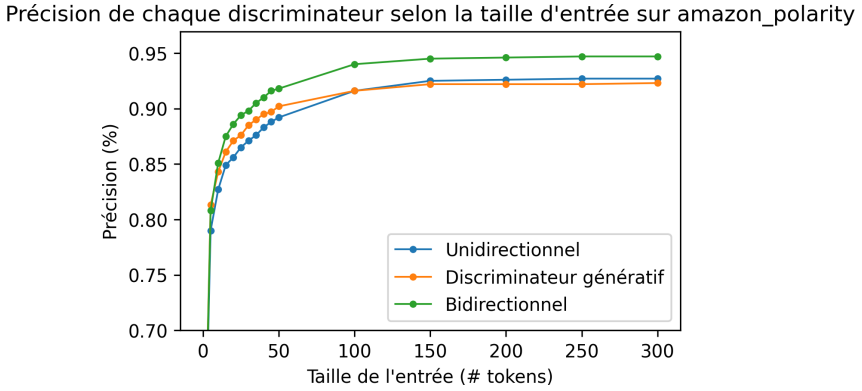


FIGURE 5 – Précision (%) des différents types de discriminateurs en fonction de la longueur de l'entrée (nb tokens) ; données amazon_polarity

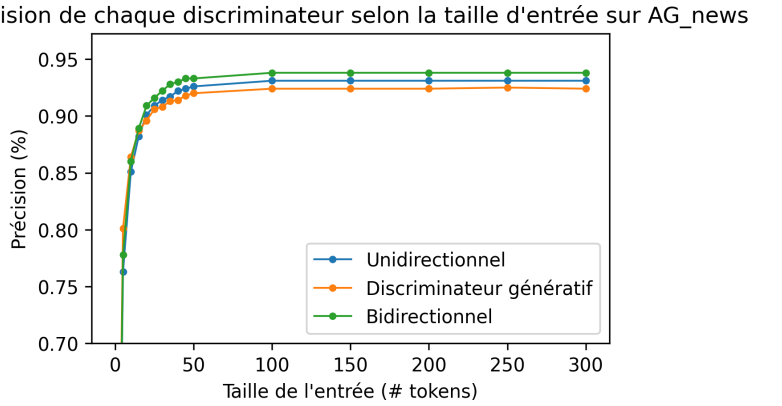


FIGURE 6 – Précision (%) des différents types de discriminateurs en fonction de la longueur de l'entrée (nb tokens) ; données AG_news

(Chaffin *et al.*, 2021) en contraignant la génération vers une classe c. C'est-à-dire que le décodage avec le MCTS a pour objectif de guider le processus de génération vers une classe désirée c en utilisant $p(c|x)$ donné par le discriminateur considéré. Pour aider à une comparaison équitable des méthodes, la longueur maximum des générations est fixée à 98 tokens (mais peuvent être plus courte si un token de fin de séquence est générée avant).

Pour évaluer cette tâche de génération coopérative, nous utilisons :

1. la précision oracle, mesurée comme le taux moyen de séquences générées vers une classe c à être correctement classées c par un discriminateur oracle entraîné sur des données disjointes ;
2. Self-BLEU, qui se concentre sur la diversité entre les échantillons, en mesurant le score BLEU (Zhu *et al.*, 2018) entre les séquences générées ;
3. la perplexité oracle, mesurée par un LM entraîné sur des données disjointes, permettant d'estimer la qualité d'écriture des textes générés.

Ces mesures sont automatiques, au sens où elles sont calculées sans intervention humaine, mais des travaux antérieurs ont montré leur corrélation avec des évaluations humaines évaluant le respect de la

Modèle	Précision oracle ↑	5 - Self-BLEU ↓	Perplexité oracle ↓
		amazon_polarity	
$p(x)$	70,8	0,652	**10,49**
bidirectional	**96,0***	0,531*	12,25
unidirectional	93,0*	0,528*	11,98
unidirectional (100 its)	93,6*	**0,522***	10,73
génératif	84,4	0,576	11,92
		AG_news	
$p(x)$	86,6	**0,306**	**29,08**
bidirectional	**94,8***	0,319	29,13
unidirectional	93,4	0,313	29,99
unidirectional (100 its)	94,6*	0,323	30,92
génératif	91,8	0,321	29,43

TABLE 1 – Performances du décodage sur amazon_polarity et AG_news. $*$ indique une amélioration statistiquement significative par rapport au discriminateur génératif. Aucun modèle n'a démontré une amélioration significative par rapport au discriminateur unidirectionnel.

contrainte (précision) et la qualité de génération (perplexité) (Chaffin *et al.*, 2021).

Pour obtenir l'évaluation la plus précise, le discriminateur oracle est un modèle BERT bidirectionnel. Afin d'utiliser la même tokénisation, les modèles de langue sont également des modèles BERT avec une tête LM (à la différence de des expériences de Chaffin *et al.* (2021) qui reposait sur un modèle GPT-2). Pour cette expérience, nous utilisons les hyper-paramètres les plus performants de la littérature ($c_{puct} = 3$, température $\tau = 1$) et 50 itérations du MCTS par token, sauf indication contraire.

Les résultats moyennés sur 500 textes générés en utilisant chaque type de discriminateur sur les deux jeux de données sont rapportés en Table 1, avec leur significativité statistique (t-test, $p = 0,01$). Nous présentons également les résultats du MCTS guidé par la vraisemblance du LM non modifié $p(x)$, afin de fournir des résultats atteignables sans discriminateur.

Ces résultats montrent que la différence de précision brute entre discriminateurs bi- et unidirectionnels se reflète sur la qualité des générations qu'ils guident (cf. précision oracle), mais de manière limitée. De plus, doubler le nombre d'itérations du MCTS (100 its) permet d'augmenter les résultats de précision du modèle unidirectionnel, comblant ainsi l'écart entre les deux modèles pour un coût de calcul qui reste plus faible (voir section suivante). Les résultats utilisant des discriminateurs génératifs sont différents, avec une baisse de précision significativement plus importante qu'entre les modèles unidirectionnels et bidirectionnels sur AG_news, bien que l'écart dans la précision brute soit similaire. Sur amazon_polarity, malgré des précisions brutes similaires, nous observons une baisse de la précision oracle de 10 points. Cela s'explique par le fait que le signal de GeDi n'est pas assez informatif : à précisions brutes similaires, le score moyen attribué à la bonne classe par GeDi est nettement inférieur à celui des autres discriminateurs. Sur les autres métriques, le type de discriminateur n'a pas d'impact significatif.

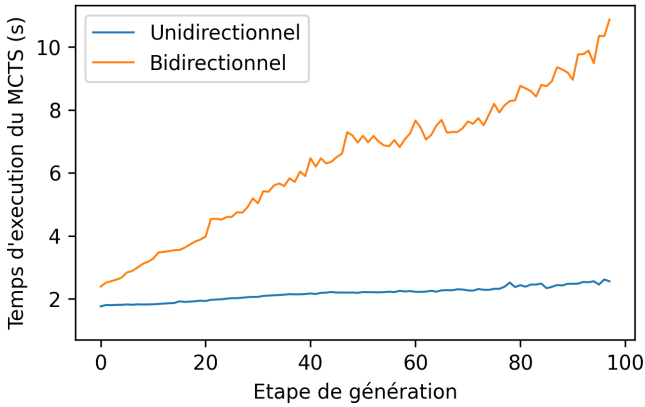


FIGURE 7 – Temps d'exécution des itérations MCTS (s) en fonction du nombre d'étapes de génération (moyenné sur 10 batchs de 30 séquences ; amazon_polarity)

3.3 Gain calculatoire

Au-delà de la qualité de génération, nous nous intéressons à la complexité de calcul des différents discriminateurs dans la génération coopérative. La figure 7 présente les temps d'exécution du MCTS pour une étape de génération t (i.e. le temps nécessaire pour décoder le token à l'étape t de n'importe quelle séquence) avec un modèle unidirectionnel et bidirectionnel. Comme attendu, puisque la complexité est quadratique dans le cas bidirectionnel et seulement linéaire dans le cas unidirectionnel, la différence de temps de génération est significative et augmente (linéairement) avec la longueur de la séquence. Le nombre d'itérations du MCTS avec un discriminateur unidirectionnel peut être plus que doublé par rapport à un bidirectionnel, tout en gardant un coût de calcul significativement plus bas, même pour de petites séquences de texte.

Dans le cas du discriminateur génératif, un grand gain de calcul est espéré du fait que les scores de discrimination sont calculés pour chaque enfant d'un nœud (séquence) en seulement $|\mathcal{C}|$ passes avant (*forward pass*). En effet, alors que le calcul des scores pour chacun des $|\mathcal{V}|$ nœuds enfants coûterait $|\mathcal{V}|$ passes avant dans le cas des classificateurs discriminants, il ne nécessite que $|\mathcal{C}|$ passes avant pour les classificateurs génératifs (c'est-à-dire une passe par classe pour obtenir tous les scores, plutôt qu'une passe par nœud enfant). Puisque généralement $|\mathcal{C}| \ll |\mathcal{V}|$, l'utilisation d'un discriminateur génératif semble a priori très avantageuse et permet, à coût calculatoire constant, d'augmenter le nombre d'itérations du MCTS pour rattraper, voire dépasser, l'écart de précision avec les approches discriminatives. Cependant, ce gain potentiel dépend fortement de l'exploration de l'arbre et donc du paramètre c_{puct} du MCTS. Si moins de $|\mathcal{C}|$ enfants sont considérés à chaque niveau de l'arbre, alors l'approche générative est au moins aussi coûteuse que l'approche discriminative et peut même être plus coûteuse.

Empiriquement, nous observons que pour les valeurs usuelles d'exploration, GeDi est beaucoup plus coûteux. Augmenter l'exploration réduit ce coût mais également la précision des textes générés. En figure 8, nous montrons l'influence de ce paramètre sur la précision (à gauche) et sur le coût (à droite).

On voit ainsi que pour la valeur usuelle $c_{puct} = 3$, les discriminateurs génératifs ont besoin en moyenne de 1 685 passes avant de plus sur amazon_polarity (où $|\mathcal{C}|$ est seulement 2), ce qui signifie

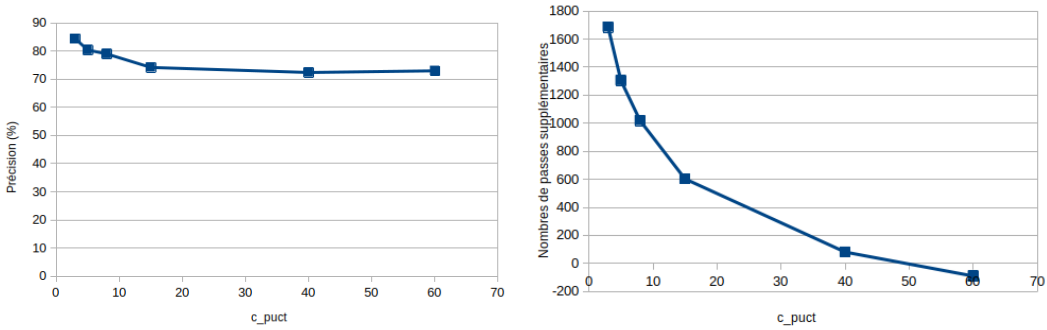


FIGURE 8 – Précision (à gauche) et nombres de passes avant supplémentaires (à droite) en fonction du paramètre c_{puct} sur les données amazon_polarity.

qu'il y a plus d'exploration en profondeur qu'en largeur. Augmenter c_{puct} diminue cette différence, mais aussi la précision de génération qui en résulte. À $c_{puct} = 15$, la précision chute déjà de 10 points, et la différence est toujours en défaveur de GeDi pour plus de 600 passes avant. Avec une valeur très élevée ($c_{puct} = 50$), le coût supplémentaire (nombre de passes avant) disparaît, mais l'impact sur la précision reste trop important pour que cette solution soit adoptée en pratique.

Ces résultats montrent que les discriminateurs génératifs ne sont bénéfiques que si l'exploration est plus large que profonde, ce qui n'est pas le cas pour l'usage du MCTS, mais peut être le cas avec d'autres stratégies de décodage. Ceci est cohérent avec les résultats de GeDi (Krause *et al.*, 2021), qui ont montré un gain important dans une approche de décodage par recherche en faisceau (*beam search*) où la largeur est cruciale. Ces nouveaux résultats suggèrent de chercher des moyens de mieux exploiter ce potentiel de GeDi, soit avec une exploration plus efficace en largeur du MCTS, soit en utilisant des méthodes qui le font par construction comme le *beam search*.

4 Conclusion

La génération coopérative s'avère être un moyen efficace d'améliorer la génération de texte traditionnelle avec des informations externes provenant d'un discriminateur. Alors que les transformers à attention bidirectionnelle sont généralement préférés pour les tâches discriminatives, ils ne sont pas auto-régressifs et sont donc beaucoup plus coûteux lorsqu'ils sont utilisés pour guider la génération.

Bien qu'un peu moins précis qu'avec l'attention bidirectionelle, les transformers unidirectionnels permettent quant à eux d'obtenir des résultats très similaires pour un coût bien plus faible dans ce contexte d'utilisation. Notre étude montre ainsi que les discriminateurs unidirectionnels sont un meilleur choix pour la génération coopérative, pour laquelle de légères baisses de précision peuvent être compensées par le réinvestissement d'une partie du gain computationnel dans l'exploration de plus de solutions.

Compte tenu de la taille des vocabulaires usuels, les discriminateurs génératifs semblaient a priori très intéressants pour permettre une recherche plus large. Cependant, les parcours d'arborescence effectués par le MCTS ne permettent pas de profiter de ces meilleures capacités d'exploration en

largeur, et l'évaluation de l'ensemble du vocabulaire se fait au prix d'un signal moins informatif. Une meilleure utilisation de cette largeur dans les approches de décodage coopératif est cependant une piste très prometteuse.

Pour permettre la réplicabilité et d'autres expériences sur ce sujet, le code utilisé pour nos expériences est disponible à https://github.com/NohTow/PPL-MCTS/tree/main/teammates.

Références

BAKHTIN A., DENG Y., GROSS S., OTT M., RANZATO M. & SZLAM A. (2021). Residual energy-based models for text. *Journal of Machine Learning Research*, **22**(40), 1–41.

BENDER E. M., GEBRU T., MCMILLAN-MAJOR A. & SHMITCHELL S. (2021). On the dangers of stochastic parrots : Can language models be too big ? In M. C. ELISH, W. ISAAC & R. S. ZEMEL, Éds., *FAccT '21 : 2021 ACM Conference on Fairness, Accountability, and Transparency, Virtual Event / Toronto, Canada, March 3-10, 2021*, p. 610–623 : ACM. DOI : 10.1145/3442188.3445922.

CACCIA M., CACCIA L., FEDUS W., LAROCHELLE H., PINEAU J. & CHARLIN L. (2020). Language GANs Falling Short. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020* : OpenReview.net.

CHAFFIN A., CLAVEAU V. & KIJAK E. (2021). PPL-MCTS : Constrained Textual Generation Through Discriminator-Guided MCTS Decoding. *CoRR*, **abs/2109.13582**.

CHAFFIN A., CLAVEAU V. & KIJAK E. (2022). Décodage guidé par un discriminateur avec le monte carlo tree search pour la génération de texte contrainte. In *Actes de la conférence TALN 2022* : ATALA.

CHEN X., CAI P., JIN P., WANG H., DAI X. & CHEN J. (2020). Adding a filter based on the discriminator to improve unconditional text generation. *arXiv preprint arXiv :2004.02135*.

CLARK E., AUGUST T., SERRANO S., HADUONG N., GURURANGAN S. & SMITH N. A. (2021). All that's 'human' is not gold : Evaluating human evaluation of generated text. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1 : Long Papers)*, p. 7282–7296, Online : Association for Computational Linguistics. DOI : 10.18653/v1/2021.acl-long.565.

CLAVEAU V. (2021). Neural text generation for query expansion in information retrieval. In *WI-IAT 2021 - 20th IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, Proceedings of the WI-IAT Conference, p. 1–8, Melbourne, Australia : IEEE. DOI : 10.1145/3486622.3493957, HAL : hal-03494692.

COULOM R. (2006). Efficient selectivity and backup operators in monte-carlo tree search. In H. J. VAN DEN HERIK, P. CIANCARINI & H. H. L. M. DONKERS, Éds., *Computers and Games, 5th International Conference, CG 2006, Turin, Italy, May 29-31, 2006. Revised Papers*, volume 4630 de *Lecture Notes in Computer Science*, p. 72–83 : Springer. DOI : 10.1007/978-3-540-75538-8_7.

DATHATHRI S., MADOTTO A., LAN J., HUNG J., FRANK E., MOLINO P., YOSINSKI J. & LIU R. (2020). Plug and play language models : A simple approach to controlled text generation. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020* : OpenReview.net.

DEPT. C.-M. U. S. (2018). Speech understanding systems : summary of results of the five-year research effort at Carnegie-Mellon University. DOI : 10.1184/R1/6609821.v1.

DEVLIN J., CHANG M., LEE K. & TOUTANOVA K. (2019). BERT : pre-training of deep bidirectional transformers for language understanding. In J. BURSTEIN, C. DORAN & T. SOLORIO, Éds., *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, p. 4171–4186 : Association for Computational Linguistics. DOI : 10.18653/v1/n19-1423.

DONG L., YANG N., WANG W., WEI F., LIU X., WANG Y., GAO J., ZHOU M. & HON H. (2019). Unified language model pre-training for natural language understanding and generation. In H. M. WALLACH, H. LAROCHELLE, A. BEYGELZIMER, F. D'ALCHÉ-BUC, E. B. FOX & R. GARNETT, Éds., *Advances in Neural Information Processing Systems 32 : Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, p. 13042–13054.

FAN A., LEWIS M. & DAUPHIN Y. N. (2018). Hierarchical neural story generation. In I. GUREVYCH & Y. MIYAO, Éds., *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1 : Long Papers*, p. 889–898 : Association for Computational Linguistics. DOI : 10.18653/v1/P18-1082.

GABRIEL S., BOSSELUT A., DA J., HOLTZMAN A., BUYS J., LO K., CELIKYILMAZ A. & CHOI Y. (2021). Discourse understanding and factual consistency in abstractive summarization. In P. MERLO, J. TIEDEMANN & R. TSARFATY, Éds., *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics : Main Volume, EACL 2021, Online, April 19 - 23, 2021*, p. 435–447 : Association for Computational Linguistics.

GEHMAN S., GURURANGAN S., SAP M., CHOI Y. & SMITH N. A. (2020). Realtoxicityprompts : Evaluating neural toxic degeneration in language models. In T. COHN, Y. HE & Y. LIU, Éds., *Findings of the Association for Computational Linguistics : EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 de *Findings of ACL*, p. 3356–3369 : Association for Computational Linguistics. DOI : 10.18653/v1/2020.findings-emnlp.301.

GOODFELLOW I. J., POUGET-ABADIE J., MIRZA M., XU B., WARDE-FARLEY D., OZAIR S., COURVILLE A. C. & BENGIO Y. (2020). Generative adversarial networks. *Commun. ACM*, **63**(11), 139–144. DOI : 10.1145/3422622.

HE D., LU H., XIA Y., QIN T., WANG L. & LIU T.-Y. (2017). Decoding with value networks for neural machine translation. *Advances in Neural Information Processing Systems*, **30**.

HOLTZMAN A., BUYS J., DU L., FORBES M. & CHOI Y. (2020). The curious case of neural text degeneration. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020* : OpenReview.net.

HOLTZMAN A., BUYS J., FORBES M., BOSSELUT A., GOLUB D. & CHOI Y. (2018). Learning to write with cooperative discriminators. In I. GUREVYCH & Y. MIYAO, Éds., *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1 : Long Papers*, p. 1638–1649 : Association for Computational Linguistics. DOI : 10.18653/v1/P18-1152.

KESKAR N. S., MCCANN B., VARSHNEY L. R., XIONG C. & SOCHER R. (2019). CTRL : A conditional transformer language model for controllable generation. *CoRR*, **abs/1909.05858**.

KRAUSE B., GOTMARE A. D., MCCANN B., KESKAR N. S., JOTY S. R., SOCHER R. & RAJANI N. F. (2021). Gedi : Generative discriminator guided sequence generation. In M. MOENS, X. HUANG, L. SPECIA & S. W. YIH, Éds., *Findings of the Association for Computational Linguistics :*

EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021, p. 4929–4952 : Association for Computational Linguistics.

LAMPRIER S., SCIALOM T., CHAFFIN A., CLAVEAU V., KIJAK E., STAIANO J. & PIWOWARSKI B. (2022). Generative cooperative networks for natural language generation. *CoRR*, **abs/2201.12320**.

LEBLOND R., ALAYRAC J., SIFRE L., PISLAR M., LESPIAU J., ANTONOGLOU I., SIMONYAN K. & VINYALS O. (2021). Machine translation decoding beyond beam search. In M. MOENS, X. HUANG, L. SPECIA & S. W. YIH, Éds., *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, p. 8410–8434 : Association for Computational Linguistics.

LEWIS P., PEREZ E., PIKTUS A., PETRONI F., KARPUKHIN V., GOYAL N., KÜTTLER H., LEWIS M., YIH W.-T., ROCKTÄSCHEL T., RIEDEL S. & KIELA D. (2020). Retrieval-augmented generation for knowledge-intensive nlp tasks. In H. LAROCHELLE, M. RANZATO, R. HADSELL, M. F. BALCAN & H. LIN, Éds., *Advances in Neural Information Processing Systems*, volume 33, p. 9459–9474 : Curran Associates, Inc.

LOSHCHILOV I. & HUTTER F. (2019). Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019* : OpenReview.net.

MUSTAR A., LAMPRIER S. & PIWOWARSKI B. (2022). On the study of transformers for query suggestion. *ACM Trans. Inf. Syst.*, **40**(1), 18 :1–18 :27. DOI : 10.1145/3470562.

NAKANO R., HILTON J., BALAJI S., WU J., OUYANG L., KIM C., HESSE C., JAIN S., KOSARAJU V., SAUNDERS W., JIANG X., COBBE K., ELOUNDOU T., KRUEGER G., BUTTON K., KNIGHT M., CHESS B. & SCHULMAN J. (2021). Webgpt : Browser-assisted question-answering with human feedback. *CoRR*, **abs/2112.09332**.

NG A. Y. & JORDAN M. I. (2001). On discriminative vs. generative classifiers : A comparison of logistic regression and naive bayes. In T. G. DIETTERICH, S. BECKER & Z. GHAHRAMANI, Éds., *Advances in Neural Information Processing Systems 14 [Neural Information Processing Systems : Natural and Synthetic, NIPS 2001, December 3-8, 2001, Vancouver, British Columbia, Canada]*, p. 841–848 : MIT Press.

PALLAGANI V. & SRIVASTAVA B. (2021). A generic dialog agent for information retrieval based on automated planning within a reinforcement learning platform. *Bridging the Gap Between AI Planning and Reinforcement Learning (PRL)*.

PASUNURU R., CELIKYILMAZ A., GALLEY M., XIONG C., ZHANG Y., BANSAL M. & GAO J. (2021). Data augmentation for abstractive query-focused multi-document summarization. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI 2021). Online*, p. 13666–13674.

RADFORD A., WU J., CHILD R., LUAN D., AMODEI D. & SUTSKEVER I. (2019). Language models are unsupervised multitask learners.

REN Z., WANG X., ZHANG N., LV X. & LI L.-J. (2017). Deep reinforcement learning-based image captioning with embedding reward. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

SCIALOM T., DRAY P., LAMPRIER S., PIWOWARSKI B. & STAIANO J. (2020). Discriminative adversarial search for abstractive summarization. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 de *Proceedings of Machine Learning Research*, p. 8555–8564 : PMLR.

SCIALOM T., DRAY P., LAMPRIER S., PIWOWARSKI B. & STAIANO J. (2021). To Beam Or Not To Beam : That is a Question of Cooperation for Language GANs. *Advances in neural information processing systems*.

SILVER D., SCHRITTWIESER J., SIMONYAN K., ANTONOGLOU I., HUANG A., GUEZ A., HUBERT T., BAKER L., LAI M., BOLTON A. *et al.* (2017). Mastering the game of go without human knowledge. *nature*, **550**(7676), 354–359.

VASWANI A., SHAZEER N., PARMAR N., USZKOREIT J., JONES L., GOMEZ A. N., KAISER L. & POLOSUKHIN I. (2017). Attention is all you need. In I. GUYON, U. VON LUXBURG, S. BENGIO, H. M. WALLACH, R. FERGUS, S. V. N. VISHWANATHAN & R. GARNETT, Éds., *Advances in Neural Information Processing Systems 30 : Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, p. 5998–6008.

WOLF T., DEBUT L., SANH V., CHAUMOND J., DELANGUE C., MOI A., CISTAC P., RAULT T., LOUF R., FUNTOWICZ M., DAVISON J., SHLEIFER S., VON PLATEN P., MA C., JERNITE Y., PLU J., XU C., SCAO T. L., GUGGER S., DRAME M., LHOEST Q. & RUSH A. M. (2020). Transformers : State-of-the-art natural language processing. In Q. LIU & D. SCHLANGEN, Éds., *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing : System Demonstrations, EMNLP 2020 - Demos, Online, November 16-20, 2020*, p. 38–45 : Association for Computational Linguistics. DOI : 10.18653/v1/2020.emnlp-demos.6.

YOGATAMA D., DYER C., LING W. & BLUNSOM P. (2017). Generative and discriminative text classification with recurrent neural networks. *CoRR*, **abs/1703.01898**.

YU L., ZHANG W., WANG J. & YU Y. (2017). Seqgan : Sequence generative adversarial nets with policy gradient. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31.

YUAN R., WANG Z. & LI W. (2021). Event graph based sentence fusion. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, p. 4075–4084.

ZHANG X., ZHAO J. J. & LECUN Y. (2015). Character-level convolutional networks for text classification. In C. CORTES, N. D. LAWRENCE, D. D. LEE, M. SUGIYAMA & R. GARNETT, Éds., *Advances in Neural Information Processing Systems 28 : Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, p. 649–657.

ZHU Y., LU S., ZHENG L., GUO J., ZHANG W., WANG J. & YU Y. (2018). Texygen : A benchmarking platform for text generation models. In K. COLLINS-THOMPSON, Q. MEI, B. D. DAVISON, Y. LIU & E. YILMAZ, Éds., *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018*, p. 1097–1100 : ACM. DOI : 10.1145/3209978.3210080.