Actes de la 29e Conférence sur le Traitement Automatique des Langues Naturelles
Avignon, France, 27 juin au 1er juillet 2022

Auto-correction dans un analyseur neuronal par transitions : un comportement factice ?

Fang Zhao
Laboratoire de Linguistique Formelle, 8, Rue Albert Einstein, 75013 Paris, France
fang.zhao@etu.u-paris.fr

RÉSUMÉ

Cette étude explore la capacité d'auto-correction dans le cas d'un analyseur neuronal par transitions. Nous définissons un oracle dynamique pour le système étudié lui apprenant à s'auto-corriger. Les performances du modèle restent identiques à celles du modèle de base, qui ne s'auto-corrige pas. En effet, il y a à peu près autant de « corrections » justes que de fautives. Les erreurs finales commises par les deux modèles sont aussi similaires. Nous montrons néanmoins que beaucoup des corrections effectuées par le modèle avec oracle dynamique coïncident avec des cas difficiles à gérer par les analyseurs automatiques. Le problème d'apprentissage d'un comportement efficace d'auto-correction retombe dans un traitement efficace de ces cas difficiles.

ABSTRACT

Self-correction in a transition-based neural parser : a spurious behaviour ?

This study explores the self-correcting capacity of a transition-based neural parser. We define a dynamic oracle for the studied system that teaches it to self-correct. The performance of the model remains the same compared to the baseline model, which does not self-correct. Indeed, there are about as many right corrections as wrong ones. The final errors made by the two models are also similar. We show that, however, many of the corrections made by the model with dynamic oracle coincide with cases that are difficult to handle by the automatic parsers. The problem of learning an efficient self-correcting behavior falls back into efficient handling of these difficult cases.

MOTS-CLÉS : auto-correction, oracle dynamique, oracle statique, intégration de tâches, multi-tâche, morpho-syntaxique, sémantique.

KEYWORDS: self-correction, dynamic oracle, static oracle, multiple tasks integration, multi-task, part of speech, semantics.

1 Introduction

Dans cet article, nous parlons d'auto-correction pour désigner l'action d'un agent modifiant le résultat de ses propres décisions antérieures. Elle se produit naturellement chez les humains dans leur traitement du langage naturel. Par exemple, en lisant une phrase du type *Garden Path* telle que « *The horse raced past the barn fell.* » (Bever, 1970), un locuteur natif de l'anglais pourrait intuitivement analyser les premiers mots de la phrases en pensant que « *raced* » est le prétérit du mot « *race* » et le considère comme verbe principal de la phrase. En réalité, il s'agit de son participe passé et le verbe principal est « *fell* ». Le locuteur peut très rapidement se rendre compte de ses erreurs, car le mot

« *fell* » ne peut s'insérer dans cette analyse. Il peut ensuite se corriger, comme illustré dans la FIGURE 1, afin d'obtenir une analyse cohérente.

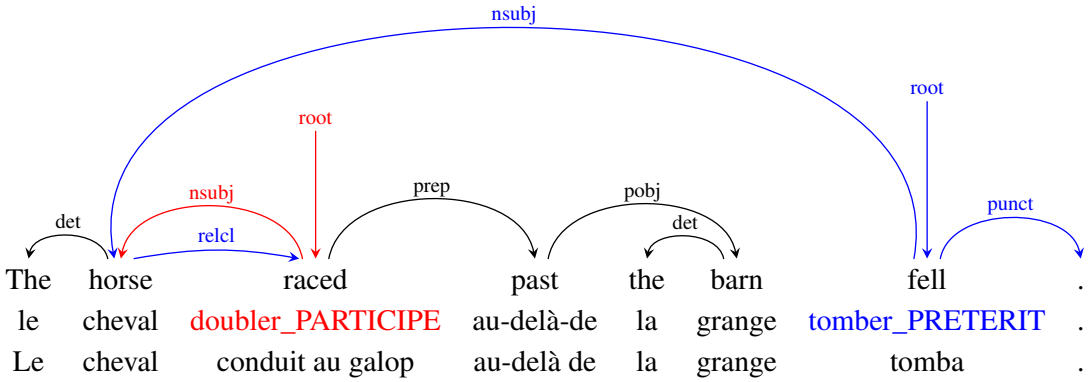

FIGURE 1 – Corrections sur l'analyse initiale de la phrase « *The horse raced past the barn fell.* ». Les dépendances rouges sont supprimées et les dépendances blues rajoutées.

Les systèmes de traitement automatique des langues (TAL) actuels ne sont, dans leur écrasante majorité, pas conçus de manière à pouvoir s'auto-corriger. De fait, l'incapacité à s'auto-corriger est particulièrement problématique pour un système prenant des décisions de manière séquentielle, où une décision à un temps t dépend des décisions précédentes : lorsqu'il effectue une erreur à un certain point, il ne peut par la suite pas la rectifier et cette erreur initiale l'amène souvent à commettre d'autres erreurs. La capacité d'auto-correction suggère donc une piste de recherche contre ce phénomène bien connu, la propagation d'erreurs (Lê & Fokkens, 2017).

Lyu *et al.* (2019) se servent de la technique du *raffinement itératif* (*iterative refinement*) pour la tâche d'étiquetage des rôles sémantiques (*Semantic Role Labeling*, *SRL*). La même technique a aussi été employée par Lee *et al.* (2018) pour la traduction automatique. Concrètement, une prédiction initiale est faite avec un modèle de base. Ensuite, un modèle de raffinement est utilisé, qui prend en entrée la phrase et la prédiction du modèle de base, et qui calcule une nouvelle prédiction. Soit cette prédiction est considérée comme finale, soit elle est réutilisée à la place de la prédiction du modèle de base pour l'itération suivante.

Nous étudions dans cet article l'auto-correction dans le cadre des tâches d'étiquetage morpho-syntaxique et d'analyse sémantique en dépendances. Plus précisément, nous partons de l'analyseur présenté par Bernard (2021), que nous nommerons *MTI-tagsynsem*. À notre connaissance, cela est le seul système par transition qui permet l'auto-correction. Contrairement à ce qui se passe dans le cas des systèmes par raffinement itératif, l'auto-correction est partie intégrante du système de transition de *MTI-tagsynsem* : durant l'analyse, il est capable de revenir à tout moment sur ses prédictions précédentes pour les modifier. Alors que Bernard (2021) a étudié l'impact d'une analyse jointe des tâches visées par rapport à une analyse séquentielle, la présente étude porte spécifiquement sur l'auto-correction réalisée en pratique par ce système.

2 Auto-correction avec un oracle dynamique

MTI-tagsynsem, comme son nom l'indique, implémente un paradigme dit d'*intégration de tâches* (*Multiple Tasks Integration*) et effectue une analyse à trois niveaux linguistiques : l'étiquetage morpho-syntaxique (*tag*) l'analyse syntaxique en dépendances (*syn*) et l'analyse sémantique en dépendances (*sem*). Nous expliquons plus bas ce qu'est la MTI. Pour une première étude sur la capacité d'auto-correction de ce système, nous choisissons uniquement deux parmi ces tâches : l'étiquetage morpho-syntaxique et l'analyse sémantique en dépendances [1]. Ce choix repose sur le fait que l'analyse sémantique est une tâche plus difficile et nous attendons donc des résultats plus intéressants en correction sur cette tâche. Nous référons au système étudié dans cet article par *MTI-tagsem*.

Une pratique courante consiste à effectuer l'analyse du texte à différents niveaux de manière séquentielle, typiquement en commençant par un étiquetage morpho-syntaxique, puis une analyse sémantique. On parle de *chaîne de traitement* (*pipeline*). Une autre possibilité est d'entraîner un système *multi-tâche* (*multitask* ; Ruder, 2017), qui mène l'analyse à différents niveaux de manière indépendante mais avec une partie de ses paramètres partagés.

L'idée principale de l'intégration de tâches est que l'analyse se fait parallèlement sur tous les niveaux modélisés sans contraintes d'ordre d'exécution. De plus, chaque décision du système repose sur l'ensemble des annotations prédites jusqu'alors. Concrètement, l'analyse se fait en plusieurs étapes : à chaque étape, *pour chaque token* de la phrase, une action est choisie depuis un ensemble d'actions (intégrant tous les niveaux d'analyse). Les actions pour le token de position i peuvent être de type « assigner l'étiquette morpho-syntaxique t à i », « ajouter une dépendance sémantique de j vers i avec l'étiquette l », et on a également l'action spéciale « ne rien faire » sur ce token. Nous détaillons toutes les actions possibles dans *MTI-tagsem* un peu plus bas dans cette section. Comme mentionné plus haut, il n'y a pas de contraintes sur l'ordre d'exécution de ces actions.

Par conception, il est possible que le système revienne à tout moment sur les prédictions effectuées précédemment dans l'analyse. En effet, si le système choisit une action qui ajoute une annotation incompatible avec une annotation précédemment prédite, celle-ci sera toujours écrasée par la nouvelle annotation [2]. Il s'agit alors là d'une correction (qu'elle soit juste ou fautive) [3].

Cependant, dans une étude préliminaire, nous avons constaté que le système ne réalisait aucune correction. Deux faits seraient conjointement à l'origine de ce phénomène : i) l'usage d'un *oracle statique* pour l'entraînement du système et ii) le système de transition, qui ne permet que des corrections très limitées en sémantique.

1. L'analyse sémantique en dépendances consiste à construire un graphe de dépendances bilexicales, celles-ci représentant en général une relation prédicat-argument. La sémantique exacte des dépendances varie cependant d'un jeu de données à un autre.

2. Pour le système décrit ici, cela revient à écraser une annotation pré-existante si celle-ci est différente de la nouvelle prédiction. Par exemple, pour une dépendance sémantique déjà annotée, si une nouvelle dépendance avec le même dépendant et le même gouverneur mais une relation différente est choisie, l'ancienne dépendance sera écrasée par celle-ci. Cependant, si l'on considérait une structure d'annotation plus contrainte, telle qu'un arbre syntaxique en dépendance, il y aurait des cas plus complexes.

3. Certaines techniques, telle que la recherche en faisceaux (*beam search*), sont capables de considérer plusieurs séquences d'actions. Cependant, il n'y a pas de tâtonnement lors de l'analyse et elles sont donc différentes de l'auto-correction étudiée dans cet article. Il peut aussi y avoir des chaînes de corrections. Nous n'avons pas étudié dans cet article ce phénomène néanmoins intéressant.

L'usage d'un oracle statique [4] Dans notre expérience, *MTI-tagsem* effectue un entraînement supervisé classique. Cet entraînement repose sur un oracle statique, qui converti les annotations de référence en séquences d'actions que le modèle sera entraîné à reproduire. Comme il n'y a pas d'erreurs dans les séquences de l'oracle, et par conséquent aucune correction, le système n'apprend jamais à faire de corrections.

En conséquence, nous remplaçons l'oracle statique par un oracle dynamique (Goldberg & Nivre, 2012). L'idée centrale de l'oracle dynamique est qu'à chaque étape d'analyse, celui-ci prend en compte la structure d'annotations actuelle ainsi que les annotations de références et calcule un ensemble d'actions possibles. Les actions possibles sont essentiellement les actions qui ajoutent les annotations de référence manquantes. Si le système de transition le permet, comme dans notre cas, l'oracle dynamique inclut aussi des actions correctives si la structure d'annotations actuelle comporte des erreurs.

Par ailleurs, l'usage d'un oracle dynamique a un autre avantage. Traditionnellement, lorsque l'on entraîne un système avec oracle statique, quelles que soient les prédictions du système, on n'applique que les actions de l'oracle. Par conséquent, le système apprend à prédire des actions en se basant uniquement sur des annotations de référence. On parle de « *teacher forcing* » pour désigner l'exposition forcée aux données de référence durant l'entraînement. L'usage du « *teacher forcing* » tend à accélérer la convergence de l'apprentissage. Cependant, durant l'inférence, le système doit se reposer sur ses propres prédictions, et ce décalage entre entraînement et inférence donne souvent lieu à des erreurs (Bengio *et al.*, 2015). En inférence, si le système effectue une prédiction incorrecte [5], il se retrouve dans une situation non vue à l'apprentissage, ce qui vraisemblablement perturbe l'inférence des actions suivantes (Goldberg & Nivre, 2012).

Une pratique courante contre ce problème est l'*échantillonage programmé* (*scheduled sampling* ; Bengio *et al.*, 2015). Il s'agit d'alterner entre les actions données par l'oracle et les décisions du système durant l'entraînement afin de s'écarter intentionnellement des situations idéales. Naturellement, il est plus approprié d'utiliser un oracle dynamique pour faire de l'échantillonnage programmé : celui-ci est capable de calculer les actions possibles à chaque étape en fonction de l'évolution de l'analyse et donne des actions correctives lorsqu'une erreur est effectuée. Suivant Goldberg & Nivre (2012), nous alternons les actions effectuées durant l'entraînement entre les actions prédites par le système et celles calculées par l'oracle dynamique selon une probabilité q que nous régularisons au cours de l'entraînement [6]. Ainsi, nous encourageons le système à explorer ses propres décisions, au lieu de lui imposer toujours les actions de l'oracle.

Le système de transition Nous détaillons ici l'inventaire des actions possibles du système *MTI-tagsem*. Pour un token (de position i dans la phrase), le système peut choisir :

— TAG-t, qui ajoute une étiquette morpho-syntaxique t ;
— SEM-j-l, qui crée une dépendance sémantique avec relation l depuis le token à position j (gouverneur) vers le token actuel à position i (dépendant) ;

4. Dans cet article, oracle statique réfère à un oracle qui se serve uniquement des exemples d'entraînement sans considérer l'état d'analyse à différentes étapes. Alors qu'oracle dynamique réfère à un oracle qui génère les transitions dynamiquement en fonction de l'évolution de l'analyse.

5. Ou s'il s'écarte tout simplement de la séquence d'actions connue (c'est-à-dire, de l'oracle) lorsqu'il y a plusieurs séquences d'actions qui mènent aux annotations de référence.

6. C'est-à-dire, avec probabilité q, les actions de l'oracle sont effectuées. En pratique, la régularisation se fait à chaque évaluation : $q = 1 - p$ où p est la mesure F1 obtenue pour la tâche d'analyse sémantique.

— TOP_PRED, qui choisit le token actuel à position i comme un *prédicat sommet* (*top predicate*) dans le graphe sémantique [7] ;
— HALT, qui n'a aucun effet sur le token i.

Comme nous l'avons mentionné au début de cette section, lorsqu'une action choisie ajoute une annotation incompatible avec une annotation existante, cette dernière est effacée. Par conséquent, la possibilité des corrections dépend de la façon dont on définit les incompatibilités. En sémantique, le seul cas où il peut y avoir une correction est lorsque l'action choisie ajoute une dépendance qui a les mêmes dépendant et gouverneur qu'une dépendance existante, mais une étiquette différente. Les possibilités d'auto-correction en sémantique sont donc très limitées. Il est en particulier impossible de supprimer une dépendance.

Afin de donner à *MTI-tagsem* une capacité d'auto-correction complète en sémantique, nous ajoutons dans l'inventaire des actions un nouveau type d'actions spéciales, [ERASE], qui a pour effet de supprimer une annotation précédemment prédite. Par exemple, l'action SEM-j[ERASE] choisie pour un token à position i supprime la dépendance sémantique (quelle que soit la relation) depuis le token à position j (gouverneur) vers le token actuel à position i (dépendant), si une telle dépendance a été prédite précédemment. Sinon, s'il n'y a pas de dépendance depuis j vers i, l'action SEM-j[ERASE] sera simplement ignorée [8].

Bien que cette action soit conçue pour répondre au besoin de l'analyse sémantique, afin que les possibilités de correction soient les mêmes dans les deux cas (dans le sens où toute annotation est supprimable), nous autorisons aussi l'usage des actions [ERASE] en étiquetage morpho-syntaxique. De plus, les actions [ERASE] sont autorisées uniquement lorsque l'oracle dynamique est utilisé (l'oracle statique ne produirait de toute manière jamais de séquences d'entraînement contenant ces actions).

Remarquons aussi que dans le cas où l'on n'utilise pas d'actions [ERASE], les corrections possibles sont faites par écrasement : au sein d'une même étape, une action nouvellement choisie écrase éventuellement une ancienne. Avec les actions [ERASE], un autre type de corrections est possible : dans une certaine étape, une annotation est supprimée pour un token ; ensuite, une nouvelle annotation (potentiellement identique à celle qui vient d'être supprimée) est rajoutée dans une étape plus tard [9].

3 Expériences

Nous avons repris les mêmes données que celles utilisées par Bernard (2021), c'est-à-dire les données anglaises de *SemEval 2015 Task 18* (Oepen *et al.*, 2015). L'ensemble d'entraînement contient 33964 phrases et l'ensemble de développement 1692. En ce qui concerne les annotations en sémantique, nous travaillons avec les graphes orienté sans cycle. Dans cette étude, nous avons repris uniquement le formalisme *DELPH-IN MRS-Derived Bi-Lexical Dependencies* (*DM* ; Oepen *et al.*, 2014). Les annotations syntaxiques sont sous la forme de dépendances de *Stanford Basic* (De Marneffe & Manning, 2008), obtenues par conversion des arbres de constituants du *Penn Treebank* (*PTB* ; Marcus *et al.*, 1993).

7. Les graphes sémantiques ont un booléen sur chaque noeud, faisant du noeud un prédicat sommet.
8. Les actions qui ne font pas de sens sont toujours ignorées : p. ex., TAG-t[ERASE] alors que le token concerné est actuellement sans étiquette prédite ; SEM-j-l alors que la dépendance à ajouter existe déjà.
9. Il est aussi possible qu'aucune nouvelle annotation soit rajoutée par la suite. Nous considérons cette suppression simple aussi une correction.

Afin de mieux tirer parti du paradigme *MTI*, Bernard (2021) effectue un entraînement principal avec l'apprentissage par renforcement, précédé d'un pré-entraînement supervisé. Cependant, pour une étude d'auto-correction avec oracle dynamique, nous limitons nos expériences et discussions sur l'apprentissage supervisé uniquement. La recherche d'hyperparamètres se fait en 20 exécutions (entraînement avec des hyperparamètres différents + évaluation) chacun pour le modèle avec oracle statique et pour le modèle avec oracle dynamique. Ensuite, nous avons choisi un ensemble d'hyperparamètres ayant la meilleure F1 en sémantique pour chaque type d'oracle. Finalement, pour chaque type d'oracle, nous effectué indépendamment trois exécutions (entraînement + évaluation) avec les mêmes (meilleurs) hyperparamètres. Sauf indication contraire, les statistiques que nous reportons ci-après sont la moyenne des trois exécutions. Les écarts types sont aussi fournis entre parenthèses.

3.1 Résultats

Nous avons constaté une distinction du comportement d'auto-correction stable des deux modèles pour toutes les exécutions : dans une évaluation sur le corpus de développement (nombre de phrases : 1692) distinct du corpus d'entraînement, le modèle entraîné avec l'oracle statique n'a fait aucune correction. Cela est attendu, car comme expliqué en SECTION 2, durant l'entraînement, l'oracle statique ne montre à aucun moment au modèle de faire la correction. Le modèle ne peut par conséquent pas apprendre à se corriger. Alors que dans le cas du modèle avec oracle dynamique, nous constatons un nombre total de 2021 (± 152) corrections. C'est-à-dire, il y a approximativement une correction par phrase.

Notre question est ainsi : pour un modèle qui s'auto-corrige et qui est entraîné en étant encouragé à explorer ses propres décisions, ses performances sur les tâches sont-elles meilleures que ceux d'un modèle qui ne s'auto-corrige pas ? Nous attendons que la réponse soit aussi affirmative. Ce n'est cependant pas le cas. Le TABLEAU 1 montre que la F1 en sémantique de nos modèles sur l'ensemble de développement est à 91.0% et à 91.2% respectivement (pour comparaison, Bernard (2021) reporte une F1 sémantique à 91.1% sur l'ensemble de test). La différence entre les deux modèles est minime sur les deux tâches. Afin de comprendre pourquoi il n'y a pas de gains en performances, nous avons conduit quelques analyses quantitatives et qualitatives sur le comportement d'auto-correction du modèle avec oracle dynamique.

	Oracle statique	Oracle dynamique
F1 étiquetage morpho-syntaxique	$97,1\%$ ($\pm 0,1\%$)	$97,1\%$ ($\pm 0,0\%$)
F1 analyse de dépendances sémantiques	$91,0\%$ ($\pm 0,1\%$)	$91,2\%$ ($\pm 0,1\%$)

TABLE 1 – F1 pour chaque tâche sur le corpus de développement

3.1.1 Détails des corrections

Nous présentons dans le TABLEAU 2 quelques statistiques sur les corrections en étiquetage morpho-syntaxique et en analyse sémantique en dépendances. Conformément à notre intuition, le taux de correction est très faible (0,5%) pour les étiquettes exactes en général, tandis que pour chaque étiquette inexacte ajoutée par le système, il y a 15,1% de chance qu'elle soit ensuite modifiée. Particulièrement, la plupart des étiquettes ajoutées dans les corrections sont exactes (14,2% parmi 15,1%). Cela confirme que les corrections effectuées par le système ne sont pas arbitraires.

Supprimées → Ajoutées		Taux de correction	Nombre de corrections	Nombre d'ajouts	
Étiquetage morpho-syntaxique					
exactes	exactes	`Sans objet`	`Sans objet`	36757 ($\pm$71)	exactes
	inexactes	0,5% ($\pm 0,2\%$)	182 ($\pm$59)		
	total	0,5% ($\pm 0,2\%$)	182 ($\pm$59)		
inexactes	exactes	14,3% ($\pm 2,8\%$)	186 ($\pm$41)	1299 ($\pm$34)	inexactes
	inexactes	0,8% ($\pm 0,5\%$)	11 ($\pm$6)		
	total	15,1% ($\pm 3,3\%$)	197 ($\pm$46)		
Analyse de dépendances sémantiques					
exactes	exactes	`Sans objet`	`Sans objet`	27481 ($\pm$139)	exactes
	inexactes	2,3% ($\pm 0,5\%$)	618 ($\pm$136)		
	total	2,3% ($\pm 0,5\%$)	618 ($\pm$136)		
inexactes	exactes	30,2% ($\pm 2,8\%$)	1019 ($\pm$117)	3374 ($\pm$74)	inexactes
	inexactes	0,2% ($\pm 0,2\%$)	5 ($\pm$5)		
	total	30,3% ($\pm 2,7\%$)	1024 ($\pm$112)		

TABLE 2 – Statistiques sur les corrections dans le modèle avec oracle dynamique. `Sans objet` correspond aux cas où l'action est ignorée (voir la SECTION 2). Le taux de correction est le nombre de corrections sur le nombre d'ajouts d'annotations. Notons que comme la correction est autorisée, un token peut être étiqueté dans une étape d'analyse et cette étiquette peut être modifiée dans une autre étape plus tard (voir la SECTION 2). Un même token peut par conséquent recevoir plusieurs ajouts d'étiquettes.

Cependant, afin de savoir si ces corrections contribuent positivement à la performance du système, il faudrait considérer les chiffres bruts dans le calcul du taux de correction [10]. Remarquons que le nombre d'étiquettes exactes ayant été modifiées en étiquettes inexactes (182) et le nombre d'étiquettes inexactes ayant été modifiées en étiquettes exactes (186) sont très proches. Autrement dit, il y a à peu près autant de corrections justes que non-justes (introduction d'erreurs). Par conséquent, le résultat a peu d'impact sur la performance.

En termes des corrections de dépendances sémantiques, le taux de correction a des tendances identiques qu'en étiquetage morpho-syntaxique. En revanche, le nombre brut de corrections depuis des dépendances inexactes vers des dépendances exactes est approximativement deux fois plus que dans le sens inverse. Cela indique que la contribution des corrections en analyse sémantique est plus importante par rapport à celle en étiquetage morpho-syntaxique.

Dans l'ensemble, nous remarquons que le modèle entraîné avec l'oracle dynamique a appris un comportement d'auto-correction juste, puisque les annotations inexactes ont beaucoup plus de chance d'être corrigées que les annotations exactes. De plus, parmi les corrections des annotations inexactes, presque toutes sont justes.

Malheureusement, vu les chiffres bruts, le fait d'avoir des corrections ne garantit pas une meilleure performance. Cela est notamment vrai dans le cas de l'étiquetage morpho-syntaxique, où les erreurs introduites contrebalancent les erreurs corrigées. Concernant l'analyse sémantique, bien qu'il y ait une prédominance de corrections justes, la performance reste proche de celle du modèle avec oracle statique, qui ne s'auto-corrige pas. Nous y reviendrons dans la SECTION 5 en examinant la

10. Afin de connaître l'exactitude des étiquettes ajoutées dans les corrections dans le cas des corrections avec des actions du type [ERASE], nous cherchons le premier ajout dans les étapes suivant la suppression. Il est extrêmement rare, mais possible que le système ne donne finalement pas d'étiquette pour un token – nous considérons qu'il s'agit d'un cas inexact.

composition des erreurs dans les prédictions des deux modèles étudiés.

4 Auto-correction en fonction des difficultés linguistiques

Dans cette section, nous présentons plusieurs analyses sur le comportement auto-correctif du modèle avec oracle dynamique. Notre principal objectif est de trouver les difficultés linguistiques derrière les corrections effectuées par le modèle. En particulier, nous nous intéressons à un type d'ambiguïté dans le langage naturel concernant l'étiquetage morpho-syntaxique et à l'effet de la longueur de dépendance sémantique sur l'analyse sémantique.

4.1 Ambiguïté des graphies

Considérons un type d'ambiguïté qui existe dans le langage naturel : certaines formes orthographiques (graphies) correspondent à différentes catégories morpho-syntaxiques. Ce type d'ambiguïté est justement problématique pour les analyseurs automatiques.

Rang	Paire d'étiquettes		Ambiguïté $n_{i,j}$	Rang corrections
1	NN	VB	14603	10
2	VBD	VBN	13179	11
3	DT	IN	12610	14
4	JJ	NN	12010	3
5	IN	RB	9859	2

(a) Les cinq paires d'étiquettes les plus ambiguës en graphie dans le corpus d'entraînement. Nombre des rangs : 169.

Rang	Paire d'étiquettes		Corrections	Rang corpus
1	NN	NNP	15	17
2	IN	RB	14	5
3	JJ	NN	13	4
4	NNP	NNPS	10	19
5	VB	VBD	10	37

(b) Les cinq paires d'étiquettes les plus corrigées par le modèle avec oracle dynamique. Nombre des rangs : 16.

TABLE 3 – Ambiguïté des graphies et corrections d'étiquettes morpho-syntaxiques. Sont présentés les résultats de l'exécution ayant la meilleure performance en étiquetage morpho-syntaxique.

Nous présentons dans le TABLEAU 3a les ambiguïtés d'étiquettes morpho-syntaxiques les plus fréquentes dans les occurrences de graphies. La valeur de chaque paire est un comptage des graphies qui sont ambiguës entre ces deux étiquettes dans l'ensemble des données d'entraînement. Concrètement, $n_{i,j}$, le comptage d'une paire d'étiquettes (t_i, t_j) [11], est défini dans l'ÉQUATION 1, où $q_{i,j}$ est le nombre de graphies qui peuvent être étiquetées t_i ou t_j, $r_{i,j}$ le nombre de ces graphies qui ont l'étiquette t_i et $r_{j,i}$ le nombre de ces graphies qui ont l'étiquette t_j ($q_{i,j} = r_{i,j} + r_{j,i}$).

11. $t_i, t_j \in \mathbb{T}$ et $t_i \neq t_j$; $\mathbb{T}$ est l'ensemble de toutes les étiquettes morpho-syntaxiques qui apparaissent dans le corpus d'entraînement.

$$n_{i,j} = \frac{r_{i,j} \times r_{j,i}}{q_{i,j}} \quad (1)$$

Intuitivement, ce comptage nous permet d'évaluer l'ambiguïté des graphies relativement à une paire d'étiquettes. Plus $n_{i,j}$ est élevé, plus la paire d'étiquettes (t_i, t_j) est ambiguës en termes de graphies.

Pour une comparaison avec le comportement auto-correctif du modèle, nous avons compté et classé les paires d'étiquettes morpho-syntaxiques entre lesquelles il se passe le plus de corrections. Nous avons tout d'abord remarqué une prédominance des corrections du type $t_i \xrightarrow{\text{TAG-}t_i\text{[ERASE]}} \varnothing \xrightarrow{\text{TAG-}t_i} t_i$, c'est-à-dire, le cas où une certaine étiquette est supprimée dans une étape d'analyse et est ensuite rajoutée dans une étape plus tard [12]. Dans cette analyse d'ambiguïté des graphies, nous ignorons ce phénomène néanmoins intéressant qui refléterait une forme d'hésitation du modèle.

Nous présentons dans le TABLEAU 3b les paires d'étiquettes les plus corrigées par le modèle avec oracle dynamique [13]. En faisant une validation croisée avec le TABLEAU 3a, nous constatons qu'il y a une corrélation entre l'ambiguïté des graphies et les corrections du modèle. En effet, parmi les exécutions, le Spearman ρ minimal est à $0,44$ (p-valeur = $1,29 \times 10^{-9}$) et le maximal à $0,51$ (p-valeur = $6,39 \times 10^{-13}$). Ces valeurs indiquent une corrélation faible ou modérée, mais bien présente (Calkins, 2005).

Une illustration de la corrélation entre l'ambiguïté des graphies et les corrections du modèle se trouve dans la FIGURE 2. Remarquons que presque toutes les données se trouvent dans le coin en haut à gauche. Cela implique que l'ambiguïté des graphies aurait un effet sur le comportement d'auto-correction du modèle.

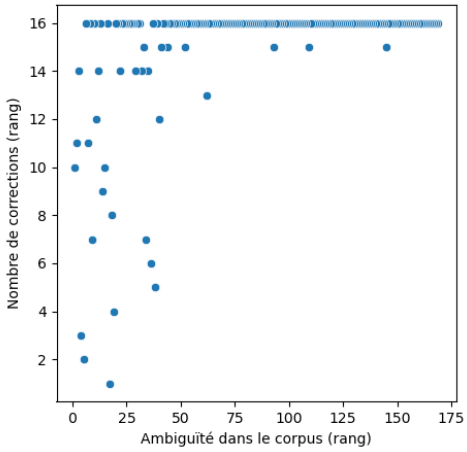

FIGURE 2 – Corrélation entre l'ambiguïté des graphies et les corrections du modèle (exécution ayant la meilleure performance en étiquetage morpho-syntaxique). Chaque point correspond à une paire d'étiquettes morpho-syntaxiques.

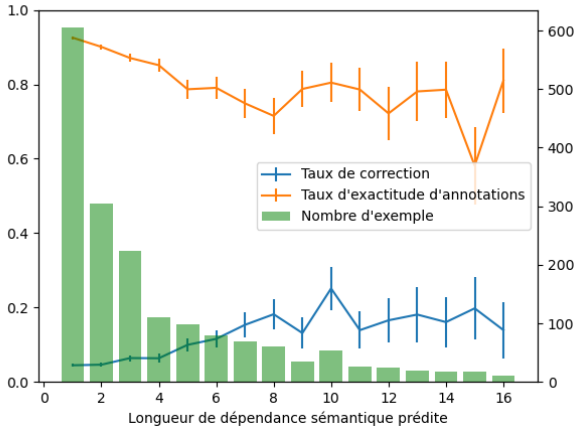

FIGURE 3 – Taux de correction et taux d'exactitude d'annotations en fonction de la longueur de dépendance sémantique prédite. Sont présentés les résultats de l'exécution ayant la meilleure performance en analyse sémantique.

12. Veuillez vous référer à la SECTION 2 sur les types de corrections possibles dans notre expérience.
13. Nous avons choisi de présenter les résultats de l'exécution ayant la meilleure performance en étiquetage morpho-syntaxique. Le nombre de corrections des paires d'étiquettes varie en fonction de l'exécution, mais comme nous allons montrer ci-dessous, la corrélation entre les corrections et l'ambiguïté dans le corpus est stable.

4.2 Longueur de dépendance sémantique

McDonald & Nivre (2007) a conduit une analyse d'erreurs en fonction de différents facteurs dont la longueur de dépendance syntaxique prédite. De la même manière, nous nous concentrons ici sur le comportement auto-correctif de notre modèle en fonction de la longueur de dépendance sémantique. Nous présentons dans la FIGURE 3 le taux de correction ainsi que le taux d'exactitude d'annotations suivant la longueur de dépendance sémantique prédite par le modèle.

Comme nous avons montré ci-dessus dans la SECTION 3.1, le taux de correction correspond au nombre de corrections par ajout d'annotation. Remarquons dans le graphe qu'en général, plus une dépendance prédite est longue, plus elle est susceptible de subir des corrections. Tandis que le taux d'exactitude d'annotations est le nombre d'annotations correctes par ajout d'annotation. Similairement, plus la dépendance prédite est longue, moins elle est susceptible d'être exacte. Dans l'ensemble, nous remarquons que la difficulté de prédiction augmente avec la longueur de dépendance sémantique et ainsi l'hésitation du modèle.

5 Cohérence des erreurs entre les modèles

Nous avons vu dans la SECTION 3 que le modèle avec oracle dynamique s'auto-corrige, bien que ces corrections ne permettent pas d'avoir de meilleures performances par rapport au modèle avec oracle statique. La question qui se pose maintenant est la suivante : si les différentes stratégies d'analyse n'ont pas d'impact sur la performance du système, est-ce qu'il y a une différence dans les prédictions, c'est-à-dire, dans les erreurs que font les deux modèles ?

Afin d'y répondre, nous proposons ici deux mesures de la cohérence des erreurs présentes dans les prédictions des deux modèles. La première concerne la cohérence des faux négatifs (c'est-à-dire, les annotations de référence qui ne sont prédites ni par l'un, ni par l'autre). La seconde concerne la cohérence des faux positifs (c'est-à-dire, les annotations incorrectes qui ont été prédites par les deux modèles).

Pour la cohérence des faux négatifs, étant données n exécutions $\mathbb{A} = \{a_1, a_2, ..., a_n\}$ d'un modèle A et m exécutions $\mathbb{B} = \{b_1, b_2, ..., b_m\}$ d'un modèle B, le score de cohérence entre le modèle A et le modèle B, $S(A, B)$, est défini dans l'ÉQUATION 2, où f est une fonction qui prend en entrée les résultats d'une exécution x du modèle X et qui renvoie l'ensemble d'annotations de référence manquantes.

$$S(A,B) = \frac{1}{2|\mathbb{A}||\mathbb{B}|} \sum_{a \in \mathbb{A}, b \in \mathbb{B}} \frac{|f(a) \cap f(b)|}{|f(b)|} + \frac{|f(a) \cap f(b)|}{|f(a)|} \tag{2}$$

Les exécutions d'un même modèle peuvent avoir des différences non négligeables dû à l'initialisation aléatoire du réseau en début d'entraînement. Afin d'avoir une base de référence pour une comparaison inter-modèles, nous proposons aussi un score de cohérence intra-modèle, $S'(A)$. En pratique, nous calculons le score de cohérence pour A et lui-même en ne prenant que des paires d'exécutions différentes, ce qui correspond à l'ÉQUATION 3.

$$S'(A) = \frac{1}{(|\mathbb{A}|-1)^2} \sum_{\substack{a_i, a_j \in \mathbb{A}^2 \\ a_i \neq a_j}} \frac{|f(a_i) \cap f(a_j)|}{|f(a_j)|} \tag{3}$$

Pour calculer la cohérence des faux positifs, nous remplaçons tout simplement f dans les ÉQUATIONS 2 et 3 par une fonction g qui renvoie l'ensemble des annotations incorrectes prédites par l'exécution x du modèle X.

Le TABLEAU 4 donne les scores de cohérence des faux négatifs et des faux positifs intra-modèles (entre les modèles avec oracle statique : *Intra-OS* ; entre les modèles avec oracle dynamique : *Intra-OD*) et inter-modèles (entre les modèles avec oracle statique et les modèles avec oracle dynamique : *OS-OD*). Ces résultats indiquent que l'usage de l'oracle dynamique n'a pas apporté de vrais changements dans le fonctionnement du système, les scores inter-modèles étant similaires aux scores intra-modèles. Il semble donc que le système n'a appris qu'un comportement d'auto-correction factice, dans le sens où ses corrections n'ont ni d'effets sur la performance du système, ni d'impacts sur les erreurs commises dans ses analyses. Néanmoins, les scores Intra-OD sont supérieurs ou égaux aux scores Intra-OS, ce qui indique que l'entraînement serait plus stable.

	Intra-OS	Intra-OD	OS-OD
	Cohérence des faux négatifs		
Étiquetage morpho-syntaxique	67,0%	67,0%	66,3%
Analyse de dépendances sémantiques	64,1%	65,0%	64,1%
	Cohérence des faux positifs		
Étiquetage morpho-syntaxique	62,5%	69,7%	67,5%
Analyse de dépendances sémantiques	50,2%	67,5%	62,5%

TABLE 4 – Cohérence des erreurs de prédictions

6 Conclusion et travaux futurs

Cette étude a exploré la capacité d'auto-correction du système *MTI-tagsem*. Bien que son système de transition lui permette de revenir sur ses prédictions précédentes, cette capacité n'est pas exploitée dans le modèle de base. Nous avons identifié deux sources potentielles de ce phénomène : i) l'oracle statique utilisé pour l'entraînement ne lui apprend pas à s'auto-corriger et ii) le système de transition ne permet que des corrections très limitées en sémantique. Nous avons donc défini un oracle dynamique pour le système *MTI-tagsem*, qui lui permet d'apprendre à s'auto-corriger. Nous avons aussi rajouté un type d'actions [ERASE] pour compléter la capacité d'auto-correction du système. Nous avons constaté que le modèle a pu effectuer des corrections à la suite de ces modifications. Malheureusement, les performances du modèle restent identiques à celles du modèle de base, qui ne s'auto-corrige pas.

Nous avons comparé les erreurs dans les prédictions des deux modèles et trouvé qu'elles étaient similaires. Cela suggère que le comportement auto-correctif du modèle avec oracle dynamique était en réalité factice. Nous avons aussi montré que le modèle avec oracle dynamique faisait beaucoup de corrections sur des cas difficiles à gérer par les analyseurs automatiques, tels que les graphies ambiguës et les dépendances longues. Le problème d'apprentissage d'un comportement efficace d'auto-correction retombe dans un traitement efficace de ces cas difficiles.

En vue de ce phénomène, nous formulons l'hypothèse suivante : le modèle est peut-être assez puissant pour apprendre à associer directement l'entrée et la sortie sans avoir à s'auto-corriger. La capacité importante du modèle est associée à la taille du réseau. Nous pensons qu'en réduisant la capacité du modèle, en réduisant sa taille, l'auto-correction fera une différence. Nous réservons cette piste d'exploration pour des travaux futurs.

Remerciements

Merci à Benoît Crabbé, Marie Candito et Yiming Liang pour leurs commentaires et l'aide apportée durant la rédaction de cet article. Merci particulièrement à Timothée Bernard pour son implication dans ce travail et son aide quant à l'implémentation. Nous tenons aussi à remercier les relecteurs anonymes pour leurs conseils avisés.

Références

BENGIO S., VINYALS O., JAITLY N. & SHAZEER N. (2015). Scheduled sampling for sequence prediction with recurrent neural networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, p. 1171–1179, Cambridge, MA, USA : MIT Press.

BERNARD T. (2021). Multiple tasks integration : Tagging, syntactic and semantic parsing as a single task. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics : Main Volume*, p. 783–794, Online : Association for Computational Linguistics.

BEVER T. (1970). *The Cognitive Basis for Linguistic Structures*, In R. HAYES, Éd., *Cognition and language development*, p. 279–362. Wiley & Sons, Inc. : New York. DOI : 10.1093/acprof :oso/9780199677139.003.0001.

CALKINS K. G. (2005). Applied statistics. [Online ; accessed 01-June-2021] URL : http ://www.andrews.edu/˜calkins/math/edrm611/edrmtoc.htm.

DE MARNEFFE M.-C. & MANNING C. D. (2008). *Stanford typed dependencies manual*. Rapport interne, Technical report, Stanford University.

GOLDBERG Y. & NIVRE J. (2012). A dynamic oracle for arc-eager dependency parsing. In *Proceedings of COLING 2012*, p. 959–976, Mumbai, India : The COLING 2012 Organizing Committee.

LÊ M. & FOKKENS A. (2017). Tackling error propagation through reinforcement learning : A case of greedy dependency parsing. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics : Volume 1, Long Papers*, p. 677–687, Valencia, Spain : Association for Computational Linguistics.

LEE J., MANSIMOV E. & CHO K. (2018). Deterministic non-autoregressive neural sequence modeling by iterative refinement. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, p. 1173–1182, Brussels, Belgium : Association for Computational Linguistics. DOI : 10.18653/v1/D18-1149.

LYU C., COHEN S. B. & TITOV I. (2019). Semantic role labeling with iterative structure refinement. 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)*, p. 1071–1082, Hong Kong, China : Association for Computational Linguistics. DOI : 10.18653/v1/D19-1099.

MARCUS M. P., MARCINKIEWICZ M. A. & SANTORINI B. (1993). Building a large annotated corpus of english : The penn treebank. *Comput. Linguist.*, **19**(2), 313–330.

MCDONALD R. & NIVRE J. (2007). Characterizing the errors of data-driven dependency parsing models. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, p. 122–131, Prague, Czech Republic : Association for Computational Linguistics.

OEPEN S., KUHLMANN M., MIYAO Y., ZEMAN D., CINKOVÁ S., FLICKINGER D., HAJIČ J. & UREŠOVÁ Z. (2015). SemEval 2015 task 18 : Broad-coverage semantic dependency parsing. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, p. 915–926, Denver, Colorado : Association for Computational Linguistics. DOI : 10.18653/v1/S15-2153.

OEPEN S., KUHLMANN M., MIYAO Y., ZEMAN D., FLICKINGER D., HAJIČ J., IVANOVA A. & ZHANG Y. (2014). SemEval 2014 task 8 : Broad-coverage semantic dependency parsing. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, p. 63–72, Dublin, Ireland : Association for Computational Linguistics. DOI : 10.3115/v1/S14-2008.

RUDER S. (2017). An overview of multi-task learning in deep neural networks. *ArXiv*, **abs/1706.05098**.