Actes de JEP-TALN-RECITAL 2024
Toulouse, France, 8 au 12 juillet 2024

Auto-correction et oracle dynamique : certains effets n'apparaissent qu'à taille réduite

Fang Zhao Timothée Bernard
Laboratoire de linguistique formelle, Université Paris Cité
fang.zhao@etu.u-paris.fr, timothee.bernard@u-paris.fr

RÉSUMÉ

Nous étudions l'effet de la capacité d'auto-correction, de l'utilisation d'un oracle dynamique et de la taille du modèle, sur la performance d'un analyseur joint (morpho)syntaxe/sémantique. Nous montrons qu'avec un modèle de taille réduite, la possibilité d'auto-correction est nuisible en sémantique mais bénéfique en syntaxe, tandis que l'utilisation d'un oracle dynamique augmente la performance en sémantique. Nous constatons également que ces effets sont souvent atténués pour des modèles de taille plus importante.

ABSTRACT

Self-correction and dynamic oracle : some effects only appear at reduced size

We study the effect of the possibility of self-correction, the use of a dynamic oracle, and model size, on the performance of a joint (morpho)syntax/semantics parser. We show that with a reduced model size, the possibility of self-correction is detrimental to semantics performance but beneficial for syntax performance, and that the use of a dynamic oracle increases semantic performance. We also find that these effects are often mitigated for larger models.

MOTS-CLÉS : auto-correction, oracle dynamique, taille des modèles, syntaxe, sémantique.

KEYWORDS: self-correction, dynamique oracle, model size, syntax, semantics.

1 Introduction

Nous avons tous connu cette situation où, en lisant une phrase, nous avons initialement mal saisi son sens. Cependant, nous sommes également capable de nous corriger rapidement lorsque nous réalisons nos erreurs. Cette capacité d'auto-correction chez les humains soulève une question intéressante : les performances de systèmes du Traitement Automatique des Langues (TAL) pourraient-elles être améliorées par l'adoption de mécanismes d'auto-correction ?

Lee *et al.* (2018) et Lyu *et al.* (2019) ont montré que le raffinement itératif, un mécanisme d'auto-correction, pouvaient augmenter la performance de systèmes de traduction automatique et d'étiquetage des rôles sémantiques (*Semantic Role Labeling*, *SRL*), respectivement. Dary (2022) a étudié un mécanisme de *retour arrière* permettant une ré-analyse partielle dans le cadre d'un analyseur incrémental et a montré que ce mécanisme augmentait la performance du système en étiquetage morpho-syntaxique et analyse syntaxique en dépendances, et ce pour différentes langues. En revanche, dans un travail précédent (Zhao, 2022), nous avons montré que les corrections n'apportaient pas de gain de performance au système de transitions plus exotique de l'analyseur joint de Bernard (2021). Nous faisons

31ème Conférence sur le Traitement Automatique des Langues Naturelles,
volume 1 : articles longs et prises de position, pages 352–361.

ici l'hypothèse que la taille du réseau de neurones utilisé (c.-à-d. le nombre de paramètres du modèle) serait liée au fait que la capacité d'auto-correction n'apporte pas toujours de gain en performance ; la capacité d'auto-correction serait un atout pour un modèle de petite taille, atout dont l'impact s'amenuiserait avec la taille du modèle.

Dans ce contexte, notre étude se penche sur l'effet de différents facteurs sur la performance du système de Bernard (2021). Plus précisément, nous étudions non seulement la liberté laissée au système d'effectuer des corrections, mais aussi le type d'oracle utilisé à l'entraînement, en plus de la variation de la taille du réseau. Nous montrons que pour un petit modèle, alors qu'un oracle dynamique peut augmenter la performance dans la plupart des cas, la possibilité d'auto-correction est bénéfique ou nuisible selon les tâches. Nous constatons également qu'en augmentant la taille des modèles, les effets sont atténués voire disparaissent.

Nous souhaitons souligner le fait que le sujet étudié nous amène à travailler avec des modèles de taille réduite par rapport à l'état de l'art et ne pouvant donc pas rivaliser en termes de performance. Ce genre de modèles est cependant utile lorsque les ressources de calcul sont limitées.

2 Revue de la littérature

On désigne par *auto-correction* d'un analyseur linguistique toute action modifiant le résultat de ses décisions antérieures. Le raffinement itératif (Lee *et al.*, 2018; Lyu *et al.*, 2019) est un mécanisme d'auto-correction : un premier module est utilisé pour produire une prédiction initiale, puis un second module calcule une nouvelle prédiction fondée sur la prédiction initiale. Cette nouvelle prédiction peut ensuite être utilisée comme entrée pour une itération ultérieure du processus de raffinement. Cette technique a été utilisée notamment en traduction automatique Lee *et al.* (2018) et en étiquetage des rôles sémantiques Lyu *et al.* (2019).

Dary (2022) introduit un analyseur incrémental par transition. Cet analyseur lit le texte brut de gauche à droite, caractère par caractère, effectuant de manière jointe les tâches de segmentation (en mots et en phrase), d'étiquetage morpho-syntaxique, de lemmatisation et d'analyse syntaxique en dépendances. Un mécanisme de correction y est implémenté sous forme de retour arrière. Concrètement, en lisant le texte, lorsque le modèle reçoit une information qu'il interprète comme incompatible avec l'analyse actuelle, le modèle retourne à une certaine position précédemment traversée et effectue de nouveau l'analyse depuis cette dernière position — mais en gardant en mémoire certaines des informations acquises entre-temps.

Bernard (2021) introduit MTI-tagsynsem, un système d'analyse jointe de la morpho-syntaxe, de la syntaxe (en dépendances) et de la sémantique (en dépendances) intégrant des possibilités d'auto-correction directement dans son système de transition. Zhao (2022) montre cependant que ce système [1] n'effectue que très peu de corrections lorsqu'il est entraîné avec un *oracle statique* standard, c.-à-d. lorsque le système est entraîné à reproduire les annotations de référence sur des trajectoires sans erreur (et ne contenant donc aucune correction). Zhao s'intéresse alors à la possibilité de substituer l'oracle statique par un *oracle dynamique* (Goldberg & Nivre, 2012). Concrètement, un oracle dynamique calcule à chaque étape quelles sont les actions admissibles — dans notre cas, les actions qui introduisent des prédictions correctes et/ou corrigent des prédictions incorrectes — et si les

1. Ou plus précisément, une variante n'effectuant que les tâches d'étiquetage morpho-syntaxique et d'analyse sémantique en dépendances (donc, sans l'analyse syntaxique).

actions dont la probabilité est maximisée sont toujours des actions admissibles, les actions définissant la trajectoire explorée à l'entraînement peuvent être autant des actions admissibles (comme il est trivialement le cas avec un oracle statique) que des actions non-admissibles prédites par le modèle (ce qui introduit donc des erreurs dans la trajectoire, ensuite corrigées si le système de transition le permet — ce qui est le cas ici) [2]. Zhao (2022) montre que le système entraîné avec l'oracle dynamique apprend effectivement à effectuer des corrections, mais que la performance du système n'est pas significativement plus haute qu'avec un oracle statique.

Les biais inductifs d'un modèle d'apprentissage sont les hypothèses impliquées durant le processus d'apprentissage (Zhao *et al.*, 2018). Les biais inductifs ont été longtemps considérés comme nécessaires pour qu'un modèle puisse généraliser au-delà des exemples d'entraînement (Mitchell, 1980). Cependant, Bachmann *et al.* (2023) ont montré qu'un manque de biais inductifs pouvait être compensé par une augmentation de la taille du modèle. Ils montrent qu'à grandes tailles, les modèles construits autour d'un simple *perceptron multicouche* (*multilayer perceptron* ; MLP) peuvent atteindre sur certaines tâches classiques de vision (telles que CIFAR10 et CIFAR100 ; Krizhevsky, 2009) une performance similaire à ResNet18 (He *et al.*, 2016), un modèle fondé sur une architecture présentant de plus forts biais inductifs adaptés aux tâches de vision. Dans le même ordre d'idée, nous avançons l'hypothèse que l'absence de gain de performance observée en TAL par Zhao (2022) entre un système qui s'auto-corrige et une version sans auto-correction peut être attribuée à la taille du réseau utilisé, suffisamment importante pour atténuer les effets des possibilités d'auto-correction, et qu'un gain de performance pourrait être observé à taille plus faible.

3 Données

Nous travaillons avec une partie des données anglaises du jeu de données *SemEval 2015 Task 18* (Oepen *et al.*, 2015). Ce jeu de données contient des annotations (morpho-)syntaxiques et sémantiques notamment pour les textes du *Penn Treebank* (*PTB* ; Marcus *et al.*, 1993). Les annotations en morpho-syntaxe sont directement celles du du PTB. Pour la syntaxe, nous utilisons les arbres obtenus par conversion en dépendances *Stanford Basic* (De Marneffe & Manning, 2008) des arbres de constituants du PTB [3]. Les annotations sémantiques, le cœur de ce jeu de données centré sur l'analyse sémantique en dépendances (*Semantic Dependency Parsing*, *SDP*), sont des graphes acycliques orientés (*Directed Acyclic Graph*, *DAG*). Les noeuds de ces graphes correspondent aux tokens de la phrase et reçoivent non pas nécessairement un unique arc entrant, mais possiblement zéro ou plusieurs. En outre, chaque phrase peut avoir de zéro à plusieurs noeuds annotés comme *prédicat sommet* (*top predicate*) [4]. Alors que le jeu de données de SemEval 2015 Task 18 propose pour chaque phrase jusqu'à trois graphes sémantiques, issus de trois formalismes distincts, nous nous servons uniquement des annotations en *DELPH-IN MRS-Derived Bi-Lexical Dependencies* (*DM* ; Oepen *et al.*, 2014) dans nos expériences.

2. L'idée d'explorer ainsi des trajectoires imparfaites a ensuite été reprise pour définir l'*échantillonage programmé* (*scheduled sampling* ; Bengio *et al.*, 2015).

3. Ce jeu de données contient 33964 et 1692 phrases respectivement pour les ensembles d'entraînement et de développement. Nous réservons l'ensemble de test pour d'éventuelles études futures visant l'état de l'art, ce qui n'est pas l'objectif de cette étude.

4. La notion de prédicat sommet dans un graphe sémantique est comparable à celle de racine dans un arbre syntaxique. Sa définition précise varie d'un formalisme à l'autre (Oepen *et al.*, 2015).

4 Modèle

Nous décrivons ici la variante du système MTI-tagsynsem que nous utilisons dans cette étude. Contrairement à un système par transition usuel, ce système n'analyse pas son entrée de gauche à droite et ne repose pas sur une *pile* (*stack*) ni un *tampon* (*buffer*). À chaque étape de calcul (c.-à-d., à chaque transition), *pour chaque token* de la phrase, une action est choisie depuis un ensemble d'actions intégrant toutes les tâches (étiquetage morpho-syntaxique, analyse syntaxique en dépendances, analyse sémantique en dépendances) ; ces actions sont principalement des actions d'étiquetage ou de sélection de tête (Zhang *et al.*, 2017).

Nous détaillons ici le répertoire d'actions utilisé dans cette étude. Afin de faciliter la discussion, nous utilisons les notations $j \xrightarrow{l} i$ pour désigner une dépendance (syntaxique ou sémantique) de relation l depuis j (gouverneur) vers i (dépendant), et $j \rightarrow i$ pour désigner une dépendance de relation quelconque depuis j vers i. Pour un token (de position) i dans la phrase, les actions possibles sont :

- TAG-t, qui ajoute l'étiquette morpho-syntaxique t au token i, et TAG[ERASE], qui supprime l'étiquette de i si elle existe ;
- SYN-j-l, qui ajoute la dépendance syntaxique $j \xrightarrow{l} i$, et SYN-j[ERASE], qui supprime la dépendance syntaxique $j \rightarrow i$ si elle existe ;
- ROOT, qui détermine le token i comme étant la racine syntaxique, et ROOT[ERASE], qui détermine i comme n'étant pas la racine ;
- SEM-j-l, qui ajoute la dépendance sémantique $j \xrightarrow{l} i$, et SEM-j[ERASE], qui supprime la dépendance sémantique $j \rightarrow i$ si elle existe ;
- TOP_PRED, qui détermine le token i comme étant un *prédicat sommet* (*top predicate*), et TOP_PRED[ERASE], qui détermine i comme n'étant pas un prédicat sommet ;
- HALT qui n'a pas d'autre effet que de provoquer la fin de l'analyse lorsque cette action est simultanément choisie pour chaque token [5].

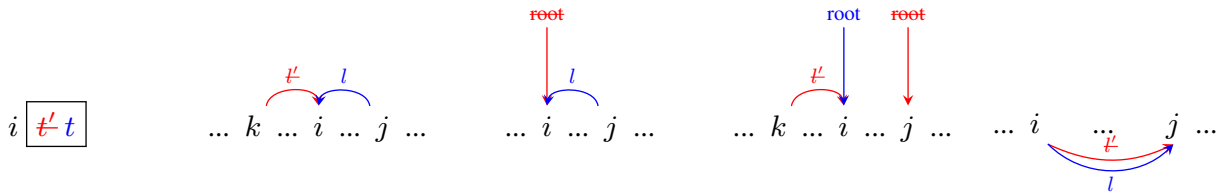
Si le système choisit pour un token une action ayant pour effet l'ajout d'une annotation alors que celle-ci existe déjà, ou la suppression d'une annotation alors que celle-ci n'existe pas, l'effet de cette action est nul. Et si une action choisie par le système ajoute une annotation incompatible avec des annotations antérieures, celles-ci sont effacées [6]. Nous détaillons ci-dessous toutes les incompatibilités dans le système utilisé. Pour un token (de position) i dans la phrase, lorsque :

a. une action TAG-t est choisie, si une étiquette morpho-syntaxique $t' \neq t$ a été prédite pour i, alors cette prédiction est supprimée ;
b. une action SYN-j-l est choisie, si 1) une dépendance syntaxique $k \rightarrow i$ a été prédite ou si 2) i a été prédit comme racine syntaxique, alors cette prédiction est supprimée ;
c. une action ROOT est choisie, si une dépendance syntaxique $k \rightarrow i$ a été prédite et/ou si un token $j \neq i$ a été prédit comme racine syntaxique, alors ces prédictions sont supprimées ;
d. une action SEM-j-l est choisie, si une dépendance sémantique $j \xrightarrow{l'} i$ avec $l' \neq l$ a été prédite, alors cette prédiction est supprimée.

Nous illustrons ces incompatibilités avec les schémas en figure 1.

5. L'analyse est aussi arrêtée automatiquement après un certain nombre d'étapes. Dans tous les cas, aucun mécanisme ne vérifie que les structures d'annotations prédites (séquence d'étiquettes morpho-syntaxiques et arbre de dépendances syntaxiques) sont complètes au moment de l'arrêt.

6. Cela est aussi le cas lorsque le système choisit des actions incompatibles entre elles au sein d'une même étape. En effet, les actions choisies pour les différents tokens lors d'une même étape ne sont pas exécutées en même temps, mais dans un ordre aléatoire. Si deux d'entre elles sont en conflit, la dernière effacera l'annotation ajoutée par la première.



a. TAG-t b.1) SYN-j-l b.2) SYN-j-l c. ROOT d. SEM-j-l

FIGURE 1 – **Illustrations des cas d'incompatibilité.** Une boîte à droite d'un token indique une étiquette morpho-syntaxique, les annotations au dessus du texte correspondent à la structure syntaxique et les annotations en dessous du texte à la structure sémantique. Les ~~annotations supprimées~~ durant l'étape illustrée sont marquées en rouge et avec une ligne de suppression. Les annotations ajoutées durant l'étape illustrée sont marquées en bleue.

Ce système nous intéresse parce qu'il intègre des possibilités d'auto-correction de manière particulièrement légère dans le sens où (i) il n'existe pas de distinctions entre deux types de phases de prédiction comme avec le raffinement itératif et (ii) la définition d'un oracle dynamique est simple, ce qui permet d'effectuer un apprentissage classique, sans nécessité de recourir à de l'apprentissage par renforcement comme avec le système de Dary *et al.* (2022). De plus, le fait que ce système effectue de manière jointe différentes tâches classiques du TAL permet d'étudier l'incidence de la capacité d'auto-correction avec plus de globalité.

5 Incidence de la capacité d'auto-correction et du type d'oracle

Comme mentionné précédemment, Zhao (2022) montre qu'un modèle MTI-tagsem (une variante de MTI-tagsynsem n'effectuant pas d'analyse syntaxique) avec auto-correction a une performance similaire à celle d'une version sans auto-correction, c.-à-d. ne différant que sur le fait que lorsqu'une action est en conflit avec des annotations déjà prédites, les effets de cette action sont ignorés. Se pourrait-il que la taille du réseau soit assez importante pour avoir compensé le gain en performance du système que l'auto-correction aurait pu apporter ? Afin de répondre à cette question, nous étudions deux séries de modèles, se distinguant par leur taille : GRAND avec $\approx 2{,}7 \times 10^6$ et PETIT avec $\approx 1{,}7 \times 10^6$ paramètres entraînables [7].

Comme Zhao (2022), nous étudions deux modèles : un premier modèle sans capacité d'auto-correction et entraîné avec oracle statique (OS-CORR) et un second modèle avec auto-correction et donc entraîné avec oracle dynamique (OD+CORR). De plus, afin de nous assurer que la différence en performance des deux modèles — si une telle différence apparaît — vient de l'auto-correction et non pas de l'oracle dynamique, nous ajoutons dans la comparaison un troisième modèle sans auto-correction mais entraîné avec oracle dynamique (OD-CORR) [8]. Notre hypothèse est qu'en dessous d'une certaine taille de modèle, les effets de la capacité d'auto-correction commencent à apparaître : le modèle avec auto-correction (OD+CORR) aura une performance plus élevée que celle des modèles sans

7. Ce comptage exclut les représentations lexicales vectorielles (*word embeddings*) pré-entraînées. Nous utilisons les vecteurs *GloVe* 6B (Pennington *et al.*, 2014) de dimension 100 et ne conservons que les vecteurs des mots apparaissant dans l'ensemble d'entraînement. Ces représentations lexicales représentent $\approx 3 \times 10^6$ paramètres entraînables supplémentaires.

8. L'oracle dynamique de ce modèle ne calcule comme actions admissibles que les actions non-correctives, c.-à-d., qui sont compatibles avec toutes les annotations actuelles.

auto-correction (OS-CORR et OD-CORR).

	Taille	OS-CORR	OD-CORR	OD+CORR
SEM	PETIT	$0{,}870_{(0{,}903\,/\,0{,}839)}$ / 0,929	$0{,}891_{(0{,}893\,/\,0{,}889)}$ / 0,995	$0{,}888_{(0{,}904\,/\,0{,}871)}$ / 0,964
	GRAND	$0{,}897_{(0{,}902\,/\,0{,}891)}$ / 0,988	$0{,}900_{(0{,}901\,/\,0{,}900)}$ / 0,998	$0{,}893_{(0{,}905\,/\,0{,}882)}$ / 0,974
SYN	PETIT	$0{,}899_{(0{,}899\,/\,0{,}899)}$ / 0,999	$0{,}899_{(0{,}899\,/\,0{,}899)}$ / 0,999	$0{,}906_{(0{,}913\,/\,0{,}900)}$ / 0,986
	GRAND	$0{,}909_{(0{,}909\,/\,0{,}909)}$ / 1,000	$0{,}907_{(0{,}907\,/\,0{,}907)}$ / 1,000	$0{,}910_{(0{,}915\,/\,0{,}905)}$ / 0,989
TAG	PETIT	$0{,}969_{(0{,}969\,/\,0{,}970)}$ / 1,000	$0{,}969_{(0{,}969\,/\,0{,}969)}$ / 1,000	$0{,}968_{(0{,}970\,/\,0{,}967)}$ / 0,997
	GRAND	$0{,}970_{(0{,}970\,/\,0{,}970)}$ / 1,000	$0{,}970_{(0{,}970\,/\,0{,}970)}$ / 1,000	$0{,}970_{(0{,}971\,/\,0{,}969)}$ / 0,997

TABLE 1 – **Performance en F1(précision / rappel) / taux d'analyse.** SEM : analyse sémantique en dépendances, SYN : analyse syntaxique en dépendances, TAG : étiquetage morpho-syntaxique. Les résultats sont calculés sur la moyenne de neuf exécutions pour chaque modèle, évalué sur l'ensemble de développement.

	Taille	1ères annotations	nombre de succès | raté | échec		
SEM	PETIT	$0{,}884_{(0{,}888\,/\,0{,}881)}$ / 0,991	$3193_{(56{,}2\%)}$	$7_{(0{,}1\%)}$	$2486_{(43{,}7\%)}$
	GRAND	$0{,}889_{(0{,}886\,/\,0{,}892)}$ / 1,006	$3086_{(56{,}6\%)}$	$9_{(0{,}2\%)}$	$2353_{(43{,}2\%)}$
SYN	PETIT	$0{,}899_{(0{,}901\,/\,0{,}897)}$ / 0,996	$3607_{(59{,}0\%)}$	$128_{(2{,}1\%)}$	$2382_{(38{,}9\%)}$
	GRAND	$0{,}903_{(0{,}904\,/\,0{,}901)}$ / 0,997	$2945_{(59{,}1\%)}$	$105_{(2{,}1\%)}$	$1929_{(38{,}7\%)}$
TAG	PETIT	$0{,}967_{(0{,}968\,/\,0{,}967)}$ / 0,999	$918_{(54{,}7\%)}$	$8_{(0{,}5\%)}$	$751_{(44{,}8\%)}$
	GRAND	$0{,}969_{(0{,}969\,/\,0{,}969)}$ / 0,999	$900_{(54{,}3\%)}$	$11_{(0{,}7\%)}$	$745_{(45{,}0\%)}$

TABLE 2 – **Analyse de corrections pour le modèle OD+CORR**. SEM : analyse sémantique en dépendances, SYN : analyse syntaxique en dépendances, TAG : étiquetage morpho-syntaxique. L'évaluation des 1ères annotations : F1(précision / rappel) / taux d'analyse. Les résultats sont calculés sur la moyenne de neuf exécutions, évalués sur l'ensemble de développement.

Le tableau 1 regroupe les performances des modèles OS-CORR, OD-CORR, OD+CORR en tailles PETIT et GRAND sur les trois tâches : analyse sémantique en dépendances, analyse syntaxique en dépendances et étiquetage morpho-syntaxique. Le système MTI-tagsynsem n'étant contraint à produire une analyse complète pour aucune des tâches (voir note 5), nous utilisons comme métriques la précision, le rappel et la F1. Nous calculons aussi, pour chaque tâche, un *taux d'analyse* : il s'agit du ratio entre le nombre d'annotations prédites et le nombre d'annotations de référence. Le taux d'analyse est toujours inférieur ou égale à 1,0 en syntaxe et morpho-syntaxe, mais peut aussi être supérieur à 1.0 en sémantique. Lorsque le taux d'analyse s'approche de 1.0, précision, rappel et F1 convergent — vers le score d'attachement étiqueté (*LAS* ou *labeled attachment score*) en syntaxe, et vers l'exactitude en morpho-syntaxe.

Nous regardons tout d'abord les résultats de l'analyse sémantique en dépendances, qui est choisie comme la tâche principale. Cela veut dire que nous utilisons la F1 en sémantique comme critère pour déterminer les meilleurs hyperparamètres et pour arrêter l'entraînement du système par arrêt précoce (*early-stopping*). En taille PETITE, comme attendu, le modèle avec oracle dynamique et auto-correction a une F1 sémantique plus élevée (0,888) que le modèle avec oracle statique et sans auto-correction (0,870). Cependant, contrairement à notre prédiction, c'est le modèle avec oracle dynamique mais sans auto-correction qui a la meilleure F1 des trois modèles (0,891). Le gain du modèle OD-CORR est principalement dû au fait que son rappel (0,893) et son taux d'analyse (0,995) sont plus élevés, bien que sa précision soit plus faible (0,893). Cela veut dire que OD-CORR a tendance à effectuer des analyses plus complètes mais au prix de la précision. Cependant, la différence de F1 en sémantique entre OS-CORR et OD-CORR est atténuée en taille GRAND.

La performance en syntaxe des modèles sans auto-correction, OS-CORR et OD-CORR, est similaire pour les deux tailles considérées. En revanche, le modèle avec auto-correction, OD+CORR, a une meilleure F1 en syntaxe que les modèles sans auto-correction quand les modèles sont petits. Cette différence s'atténue elle aussi quand les modèles sont en taille GRAND.

En termes d'étiquetage morpho-syntaxique, nous constatons que tous les modèles ont une performance similaire quelle que soit leur taille, sur toutes les métriques d'évaluation.

En résumé, l'utilisation de l'oracle dynamique augmente la performance en sémantique, alors que la capacité d'auto-correction la fait baisser. La capacité d'auto-correction fait par contre augmenter la performance en syntaxe, tandis que l'oracle dynamique seul n'a pas d'effet dans ce cas. Dans la plupart des cas, ces effets sont plus visibles lorsque les modèles ont une taille relativement petite et s'atténuent à plus grandes tailles.

Afin de comprendre comment les corrections auraient entraîné les effets en syntaxe et en sémantique mentionnés ci-dessus, nous nous proposons d'effectuer une analyse de corrections dans la section suivante.

6 Analyse de corrections

Nous classons les actions effectuées durant un épisode d'analyse de la manière suivante :

— une action est *corrective* si et seulement si elle efface au moins une annotation précédente ;
— sinon, elle est *non-corrective*.

Une action corrective peut être du type :

— *succès* si la ou les annotations supprimées étaient incorrectes, et que l'annotation éventuellement ajoutée est correcte ;
— *raté* si la ou les annotations supprimées étaient incorrectes, et que l'annotation ajoutée est aussi incorrecte ;
— *échec* si au moins une des annotations supprimées était correcte.

Nous définissons aussi : une *première annotation* est une annotation qui, lorsqu'elle est ajoutée, est compatible avec l'ensemble des premières annotations précédentes [9]. Il est possible d'évaluer les premières annotations d'un modèle avec les mêmes métriques que celles utilisées pour les prédictions globales du modèle (c.-à-d. à l'issue de l'analyse, en prenant en compte les corrections) : F1, précision, rappel et taux d'analyse.

Nous présentons les performances des premières annotations ainsi que le nombre des différents types d'actions correctives du modèle OD+CORR dans le tableau 2.

Globalement, nous constatons plus de succès que d'échecs pour toutes les tâches et tailles de modèle. En sémantique, la performance de OD+CORR en termes de premières annotations est inférieure à la performance de OD-CORR, pour toutes les métriques d'évaluation. Les actions correctives, bien que plus souvent des succès que des échecs, ne permettent pas à la performance de OD+CORR de dépasser celle de OD-CORR (indiquée en tableau 1). Nous constatons aussi que le taux d'analyse des premières annotations est proche de 1,0. Quand les corrections sont prises en compte, le taux

9. Toutes les actions non-correctives n'introduisent pas une première annotation. Considérons par exemple, pour un token de position i, que la séquence d'action choisie soit TAG-t (étape 0), TAG[ERASE] (étape 1), TAG-t' (étape 2) ; seule l'étiquetage de i avec t est considérée comme une première annotation, alors que TAG-t' en étape 2 est une action non-corrective (l'étiquetage précédent ayant été supprimée à l'étape 1).

d'analyse ainsi que le rappel baisse, mais la précision augmente.

En syntaxe, nous constatons que la performance de OD+CORR en termes de premières annotations est déjà au même niveau que la performance de OD-CORR. Compte tenu du fait qu'il y a plus d'actions correctives succès que d'échecs, la performance de OD+CORR dépasse celle de OD-CORR. Avec les corrections, par rapport aux premières annotations, la précision augmente sans que le rappel ne baisse et ce alors même que le taux d'analyse baisse.

Nous observons donc que les corrections, par rapport aux premières annotations, mènent à une baisse variable du taux d'analyse en syntaxe et en sémantique, et à une augmentation de la précision. Cela suggère que l'auto-correction a tendance à favoriser les annotations dont le modèle est plus « sûr » et de supprimer celles dont il est moins « sûr ». Cette différence entre syntaxe et sémantique est possiblement liée au fait que l'analyse sémantique est plus complexe, ne serait-ce que parce que le nombre de dépendances sémantiques, au contraire du nombre de dépendances syntaxiques, n'est pas exactement déterminé par le nombre de tokens.

Les modèles avec auto-correction effectuent beaucoup moins de corrections de l'étiquetage morpho-syntaxique que des structures syntaxiques et sémantiques. Les nombres de succès et d'échecs étant proches, la capacité d'auto-correction n'a donc quasiment aucun impact sur cette tâche.

7 Conclusion

Dans cette étude, nous avons exploré l'effet de différents facteurs — la capacité d'auto-correction, l'utilisation d'un oracle dynamique et la taille du modèle — sur la performance du système MTI-tagsynsem. Nous avons montré que les corrections étaient nuisibles pour la performance en sémantique mais étaient bénéfiques pour la performance en syntaxe. Selon notre analyse, l'auto-correction tend à favoriser les annotations les plus fiables d'après le modèle, ce qui se fait au détriment de l'analyse sémantique (qui est plus difficile). D'autre part, nous avons montré que l'utilisation d'un oracle dynamique augmentait la performance en sémantique. Nous avons constaté également que ces effets étaient atténués pour des modèles de taille plus importante, ce qui tend à conforter l'idée selon laquelle il y aurait plus à attendre sur le long terme d'une augmentation de la quantité de donnée d'entraînement et de la taille des modèles que de variations fines d'architecture ou d'algorithme (Sutton, 2019) ; une telle augmentation, cependant et lorsqu'elle est possible, n'est pas ni sans coût ni sans problème (Bender *et al.*, 2021).

Remerciements

Ces travaux ont été financés par une bourse Émergence 2021 (projet SYSNEULING) de l'IdEx *Université Paris Cité*.

Références

BACHMANN G., ANAGNOSTIDIS S. & HOFMANN T. (2023). Scaling mlps : A tale of inductive bias. *arXiv preprint arXiv :2306.13575*.

BENAMARA F., HATOUT N., MULLER P. & OZDOWSKA S., Éds. (2007). *Actes de TALN 2007 (Traitement automatique des langues naturelles)*, Toulouse. ATALA, IRIT.

BENDER E. M., GEBRU T., MCMILLAN-MAJOR A. & SHMITCHELL S. (2021). On the Dangers of Stochastic Parrots : Can Language Models Be Too Big ? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '21, p. 610–623, New York, NY, USA : Association for Computing Machinery. DOI : 10.1145/3442188.3445922.

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 P. MERLO, J. TIEDEMANN & R. TSARFATY, Éds., *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. DOI : 10.18653/v1/2021.eacl-main.66.

DARY F. (2022). *Modèles incrémentaux pour le traitement automatique des langues*. Thèse de doctorat. Thèse de doctorat dirigée par Nasr, Alexis et Fourtassi, Abdellah Informatique Aix-Marseille 2022.

DARY F., PETIT M. & NASR A. (2022). Dependency Parsing with Backtracking using Deep Reinforcement Learning. *Transactions of the Association for Computational Linguistics*, **10**, 888–903. DOI : 10.1162/tacl_a_00496.

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

DIAS G., Éd. (2015). *Actes de TALN 2015 (Traitement automatique des langues naturelles)*, Caen. ATALA, HULTECH.

GOLDBERG Y. & NIVRE J. (2012). A dynamic oracle for arc-eager dependency parsing. In *International Conference on Computational Linguistics*.

HE K., ZHANG X., REN S. & SUN J. (2016). Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, p. 770–778. DOI : 10.1109/CVPR.2016.90.

KRIZHEVSKY A. (2009). Learning multiple layers of features from tiny images. p. 32–33.

LAIGNELET M. & RIOULT F. (2009). Repérer automatiquement les segments obsolescents à l'aide d'indices sémantiques et discursifs. In A. NAZARENKO & T. POIBEAU, Éds., *Actes de TALN 2009 (Traitement automatique des langues naturelles)*, Senlis : ATALA LIPN.

LANGLAIS P. & PATRY A. (2007). Enrichissement d'un lexique bilingue par analogie. In (Benamara *et al.*, 2007), p. 101–110.

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.

MITCHELL T. M. (1980). The need for biases in learning generalizations.

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.

PENNINGTON J., SOCHER R. & MANNING C. (2014). GloVe : Global vectors for word representation. In A. MOSCHITTI, B. PANG & W. DAELEMANS, Éds., *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, p. 1532–1543, Doha, Qatar : Association for Computational Linguistics. DOI : 10.3115/v1/D14-1162.

SERETAN V. & WEHRLI E. (2007). Collocation translation based on sentence alignment and parsing. In (Benamara *et al.*, 2007), p. 401–410.

SUTTON R. S. (2019). The Bitter Lesson.

ZHANG X., CHENG J. & LAPATA M. (2017). Dependency Parsing as Head Selection. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics : Volume 1, Long Papers*, p. 665–676, Valencia, Spain : Association for Computational Linguistics.

ZHAO F. (2022). Auto-correction dans un analyseur neuronal par transitions : un comportement factice ?(self-correction in a transition-based neural parser : a spurious behaviour ?). In *Actes de la 29e Conférence sur le Traitement Automatique des Langues Naturelles. Volume 2 : 24e Rencontres Etudiants Chercheurs en Informatique pour le TAL (RECITAL)*, p. 20–32.

ZHAO S., REN H., YUAN A., SONG J., GOODMAN N. & ERMON S. (2018). Bias and generalization in deep generative models : An empirical study. *Advances in Neural Information Processing Systems*, **31**.