Actes de CORIA-TALN 2023
Paris, France, 5 au 9 juin 2023

Auto-apprentissage et renforcement pour une analyse jointe sur données disjointes : étiquetage morpho-syntaxique et analyse syntaxique

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

RÉSUMÉ

Cet article se penche sur l'utilisation de données disjointes pour entraîner un système d'analyse jointe du langage naturel. Dans cette étude exploratoire, nous entraînons un système à prédire un étiquetage morpho-syntaxique et une analyse syntaxique en dépendances à partir de phrases annotées soit pour l'une de ces tâches, soit pour l'autre. Deux méthodes sont considérées : l'auto-apprentissage et l'apprentissage par renforcement, pour lequel nous définissons une fonction de récompense encourageant le système à effectuer des prédictions même sans supervision. Nos résultats indiquent de bonnes performances dans le cas où les données disjointes sont issues d'un même domaine, mais sont moins satisfaisants dans le cas contraire. Nous identifions des limitations de notre implémentation actuelle et proposons en conséquence des pistes d'amélioration.

ABSTRACT

Self-training and reinforcement for joint analysis on disjoint data: part-of-speech tagging and syntactic parsing

In this exploratory study, we train a system to jointly perform POS tagging and syntactic dependency parsing on sentences annotated for only the former or the latter. We consider two methods: self-training and reinforcement learning, for which we define a reward function that can reward the system even for predictions for which no supervision is available. Our results indicate good performance when the disjoint data come from the same domain but are less satisfactory otherwise. We identify limitations of our current implementation and suggest possible improvements accordingly.

MOTS-CLÉS : apprentissage semi-supervisé, apprentissage par renforcement, multi-tâche, analyse jointe, étiquetage morpho-syntaxique, analyse syntaxique en dépendances, adaptation de domaine.

KEYWORDS: semi-supervised learning, reinforcement learning, multitask, joint analysis, POS tagging, syntactic dependency parsing, domain adaptation.

1 Introduction et travaux connexes

Nous cherchons dans cet article des solutions au potentiel manque de données nécessaires à l'entraînement de systèmes d'analyse jointe du langage naturel. Alors que, dans un système multi-tâche neuronal par simple partage de paramètres (Caruana, 1997; Zhang & Yang, 2017), les sorties prédites pour les différentes tâches sont calculées de manière indépendante après l'encodage de l'entrée, un système d'analyse jointe modélise de manière non triviale l'interaction entre les différentes tâches.

Actes de la 30e Conférence sur le Traitement Automatique des Langues Naturelles (TALN), volume 2 : travaux de recherche originaux – articles courts, pages 82–90.

Contrairement à un système multi-tâche simple, un tel système nécessite donc a priori pour son entraînement des données entièrement annotées, c'est-à-dire pour lesquelles les annotations couvrent, pour chaque texte, toutes les tâches modélisées. Le nombre de corpus ainsi annotés sur plusieurs niveaux d'analyse linguistique étant restreint, nous étudions la possibilité d'utiliser des jeux de données disjoints — c'est-à-dire tels que chacun n'est muni d'annotations que sur l'un des niveaux cibles — en comparant plusieurs techniques visant à palier à l'existence d'annotations manquantes. Notons que ce problème a été abordé notamment par Peng *et al.* (2018) dans le cadre d'un système par optimisation linéaire en nombres entiers.

Notre étude se fonde sur une variante du système de Bernard (2021), capable d'*intégrer* des tâches d'étiquetages lexicales ainsi que des tâches de création de dépendances bi-lexicales, c'est-à-dire de les traiter ensemble comme une tâche unique. Il s'agit d'un système par transition ayant la particularité qu'à chaque étape, jusqu'à une action par token peut être effectuée ; à tout moment, l'action d'un token peut indifféremment se rapporter à n'importe laquelle des tâches traitées, sans contrainte particulière. Chaque décision est basée sur toutes les structures précédemment prédites, ce qui permet une interaction complète entre les différentes tâches. Ce système a été choisi pour sa flexibilité (il est en particulier possible de l'entraîner sans modification sur des jeux de données disjoints, au prix des performances) et son mode d'entraînement : pré-entraîné de manière supervisée, il a été conçu pour ensuite apprendre à exploiter au mieux les interactions entre niveaux linguistiques lors d'une phase d'*apprentissage par renforcement* (Sutton & Barto, 2018). Contrairement à l'apprentissage supervisé standard qui consiste à entraîner explicitement un modèle à reproduire des annotations connues à l'avance, il s'agit d'entraîner le modèle sans le guider a priori mais en associant une *récompense* à chacune de ses actions. L'apprentissage par renforcement nécessite donc le calcul de ces récompenses, qui se fait généralement à partir d'annotations. Il est cependant possible d'apprendre une fonction de récompense ; c'est ce que fait notamment l'algorithme *apprentissage par renforcement inverse* (Ng & Russell 2000 ; voir aussi Finn *et al.* 2017). Ici, nous testons la possibilité d'utiliser les prédictions du modèle lui-même pour calculer les récompenses de ses actions pour lesquelles nous ne disposons pas d'annotation de référence.

Nous travaillons ici avec des données en anglais pour deux tâches classiques du traitement automatique des langues : l'analyse morpho-syntaxique et l'analyse syntaxique en dépendances. Bien que les corpus annotés en syntaxe soient généralement aussi annotés en morpho-syntaxe, l'inverse n'est pas le cas. De plus, ces expériences doivent être vues comme une première étape avant d'aborder le cas de tâches plus complexes, et la forte interdépendance entre analyse syntaxique et analyse morpho-syntaxique nous intéresse ici. Depuis l'article de Li *et al.* (2011), un certain nombre de travaux se sont penchés sur l'analyse jointe de ces deux tâches, mais toujours à partir de données entièrement annotées (notamment Hatori *et al.*, 2011; Bohnet & Nivre, 2012; Alberti *et al.*, 2015; Nguyen & Verspoor, 2018; Bernard, 2021).

Nous expérimentons deux méthodes. La première se concentre sur la phase d'apprentissage par renforcement et consiste à définir une fonction de récompense visant à encourager de manière pertinente le système à effectuer des prédictions même lorsque les annotations correspondantes ne sont pas disponibles. La seconde s'applique autant à la phase de pré-entraînement qu'à la phase de renforcement et est une forme d'*auto-apprentissage* (*self-learning* ou *-training* ; Nigam & Ghani 2000 ; à rapprocher, pour les modèles génératifs, de l'*algorithme espérance-maximisation*, ou EM ; Dempster *et al.* 1977) consistant, à un temps t, à utiliser le système pour prédire au moins une partie des structures d'annotations manquantes du corpus d'entraînement et ainsi pouvoir poursuivre l'apprentissage au temps $t + 1$ sur le jeu de données ainsi complété (mais donc aussi bruité).

Nous montrons l'efficacité de ces deux méthodes dans un scénario utilisant des données disjointes créées à partir d'un même corpus. Nous testons ensuite nos méthodes dans un scénario plus réaliste, utilisant des données issues de deux corpus de domaines différents. Il s'agit d'un cas de *glissement de domaine* (*domain shift* ; Ramponi & Plank 2020; Li 2012). Les résultats dans ce cadre-là sont pour l'instant moins probants, mais nous identifions un certain nombre de limitations de notre implémentation actuelle les expliquant et que nous corrigerons par la suite.

2 Modèle

Nous décrivons ici brièvement le modèle utilisé. Il ne diffère de celui de Bernard (2021) que par les tâches traitées (analyse morpho-syntaxique et analyse syntaxique en dépendances, sans analyse sémantique) et le processus d'entraînement (voir sections suivantes).

Le système fait intervenir quatre types d'actions. Pour un token de position i, effectuer (a) TAG-t consiste à assigner l'étiquette morpho-syntaxique t à i, (b) SYN-j-l à ajouter une dépendance syntaxique d'étiquette l de j vers i, (c) ROOT à définir i comme la racine de l'arbre syntaxique, et (d) HALT à ne rien faire.

À chaque étape de l'analyse d'une phrase, le système encode la phrase elle-même [1] ainsi que les prédictions effectuées aux étapes précédentes sous forme d'un vecteur par token. Chacun de ces encodages est converti en une distribution de probabilité sur l'ensemble des actions puis, pour chaque token, l'action de probabilité maximale est effectuée (une action par token est donc effectuée à chaque étape). Le système s'arrête lorsque pour chaque token est sélectionnée l'action HALT ; il n'est pas contraint à produire des structures d'annotations (ex : arbres syntaxiques) complètes. Cependant, en pratique, le taux d'analyse pour chaque tâche (le ratio du nombre de dépendances ou d'étiquettes prédites par rapport à celles annotées) dépasse toujours 99% lorsque l'entraînement s'effectue à partir de données entièrement annotées.

3 Données

Pour notre première expérience, nous utilisons la portion WSJ (*Wall Street Journal*) du corpus *Penn Treebank* (PTB ; Marcus *et al.* 1993). Nous utilisons comme arbres de dépendances de référence des conversions des arbres de constituants du PTB (de Marneffe *et al.*, 2006). Pour ces données, nous utilisons comme source la tâche 18 de la campagne SemEval 2015 (Oepen *et al.*, 2015), qui fournit aussi des graphes d'analyse sémantique de références, afin de pouvoir aisément mettre en place de futures expériences impliquant cette tâche plus complexe. Nous n'utilisons que les portions d'entraînement (sections 0 à 19 ; 33964 phrases) et de développement (section 20 ; 1692 phrases) : tous les résultats reportés dans ce texte sont calculés sur les données de développement ; nous réservons les données de test (section 21) pour une phase de comparaison ultérieure.

Pour étudier un scénario plus réaliste, impliquant un glissement de domaine, nous avons également démarré une expérience sur un jeu de données construit à partir de deux corpus de type *Universal Dependencies* (UD ; Nivre *et al.*, 2020), contenant eux aussi des phrases associées à des séquences

1. Des vecteurs GloVe (Pennington *et al.*, 2014) sont utilisés.

d'étiquettes morpho-syntaxiques et des arbres de dépendances syntaxiques. Le premier corpus, *Georgetown University Multilayer* (GUM ; Zeldes, 2017), comprend un douzaine de types de textes très variés, incluant des textes académiques, des billets de blog, des œuvres de fiction et des textes issus de médias sociaux. Le second corpus, *English Web Treebank* (EWT ; Silveira *et al.*, 2014), comprend des textes issus de diverses sources en ligne et regroupés en cinq catégories : *weblogs*, *newsgroups*, *emails*, *reviews* et *Yahoo ! answers*. Nous avons créé un jeu d'entraînement en sélectionnant aléatoirement 6911 phrases dans la portion d'entraînement de chacun des deux corpus pour un total de 13822 phrases, et de même avec un total de 2234 phrases pour le jeu de développement. Pour ce corpus aussi les résultats reportés sont calculés sur les données de développement.

Dans les expériences détaillées dans les sections suivantes, la configuration COMPLET correspond à un entraînement sur les données d'entraînement telles que décrites ci-dessus. Pour la configuration DISJOINT, par contre, pour une moitié des phrases d'entraînement les arbres syntaxiques sont ignorés et pour l'autre moitié les séquences d'étiquettes morpho-syntaxiques sont ignorées. Dans le cas du corpus PTB, cette division est effectuée au hasard. Dans le cas du corpus mixte GUM+EWT, nous ignorons les arbres syntaxiques des phrases issues du corpus EWT et inversement pour les séquences d'étiquettes morpho-syntaxiques.

4 Scénario idéal : corpus homogène

Rappelons que le modèle utilisé commence son entraînement par une phase de pré-entraînement, durant laquelle le système apprend à reproduire de manière supervisée les annotations de références connues. Dans la configuration DISJOINT, le système apprend donc à produire, pour chaque phrase, soit un arbre syntaxique, soit une séquence d'étiquettes morpho-syntaxiques, mais jamais les deux. Pour améliorer cette situation, nous implémentons un processus d'auto-apprentissage : trois fois par époque, le corpus d'entraînement est analysé par le système et les prédictions obtenues sont utilisées pour compléter les annotations et ainsi servir à l'apprentissage du système lui-même.

Après la phase de pré-entraînement commence la phase d'apprentissage par renforcement. Durant cette phase, le système analyse librement les phrases qu'on lui présente et est entraîné avec l'algorithme REINFORCE (Williams, 1992) à maximiser des *récompenses* associées à chaque action effectuée : positives pour les actions correspondant aux annotations de références, négatives pour celles contraires à celles-ci, nulles sinon (typiquement, lorsqu'il n'y a pas d'annotation correspondante). Ce système de récompenses est le système *zéro*. Un de ses effets sur notre corpus mixte GUM+EWT est qu'il n'incite pas le modèle à analyser les phrases de GUM en morpho-syntaxe ni celles de EWT en syntaxe. Afin d'encourager le système à analyser chaque phrase sur les deux niveaux linguistiques, nous implémentons le système de récompenses *auto* (pour « auto-renforcement » ; l'idée est qu'à l'issu du pré-entraînement, la plupart des prédictions du modèle sont correctes) : pour chaque token issu d'une phrase pour laquelle l'arbre syntaxique de référence n'est pas disponible, la première dépendance entrante prédite est considérée comme correcte et mène donc à une récompense positive, et de même pour les étiquettes morpho-syntaxiques [2]. Notons que l'auto-apprentissage est aussi possible durant la phase de renforcement. Cependant, lorsque l'auto-apprentissage complète entièrement les annotations, la distinction entre les systèmes *zéro* et *auto* disparaît.

2. Nous précisons « la première » car le système de transition du modèle lui permet de choisir une nouvelle dépendance entrante pour un token en ayant déjà une (effaçant alors la prédiction initiale et menant à une récompense négative d'après *auto*), et de même pour les étiquettes morpho-syntaxiques. Dans les présentes conditions, cependant, cette possibilité n'est pas utilisée de manière significative (Zhao, 2022).

La table 1 présente les performances sur le jeu de données PTB de différents modèles [3]. Les deux premières lignes concernent la configuration COMPLET et indiquent les bornes supérieures du modèle avec et sans renforcement. Les lignes suivantes concernent la configuration DISJOINT, avec cinq modèles se distinguant par un pré-entraînement simple (-AR) ou suivi d'un apprentissage par renforcement (+AR[zéro] ou +AR[auto] suivant le système de récompense utilisé) et de la complétion (+AA) ou non des annotations par auto-apprentissage (autant durant le pré-entraînement que la phase de renforcement si elle a lieu).

Données	Apprentissage	SYN	TAG	Chevauchement	Taux SYN	Taux TAG
COMPL	-AR*	0,912$_{(0,912/0,912)}$	0,971$_{(0,971/0,971)}$	1,000	1,000	1,000
	+AR$_{[zéro]}$	0,918$_{(0,919/0,918)}$	0,973$_{(0,973/0,973)}$	1,000	0,999	1,000
DISJ	-AR*	0,788$_{(0,918/0,691)}$	0,374$_{(0,940/0,235)}$	0,096	0,753	0,249
	-AR+AA	0,852$_{(0,883/0,823)}$	0,969$_{(0,969/0,969)}$	0,974	0,932	1,000
	+AR$_{[zéro]}$*	0,883$_{(0,896/0,870)}$	0,968$_{(0,969/0,967)}$	0,986	0,970	0,999
	+AR$_{[auto]}$	0,891$_{(0,891/0,891)}$	0,970$_{(0,970/0,970)}$	1,000	0,999	1,000
	+AR$_{[auto]}$+AA	0,898$_{(0,900/0,897)}$	0,971$_{(0,971/0,971)}$	0,999	0,997	1,000

TABLE 1 – Performances sur le PTB ; chaque valeur est une moyenne sur 9 exécutions. Format : F1$_{(Précision/Rappel)}$ pour la syntaxe (SYN) et la morpho-syntaxe (TAG). Pour chaque groupe de comparaison, les modèles sont comparés avec un modèle de référence (indiqué par *) et les différences significatives sont indiquées ainsi (p-valeur < 0,01 ; test de permutation de Pitman).

Nous constatons tout d'abord que dans la configuration DISJOINT, les performances du modèle -AR sont très faibles. Ce système produit des analyses plutôt justes (voir les valeurs de précision) mais très incomplètes : la colonne Taux SYN indique que seuls 75,3% des tokens se voient prédire une tête syntaxique ou le statut de racine, et la colonne Taux TAG indique que 24,9% des tokens se voient prédire une étiquette morpho-syntaxique. Le taux de chevauchement (un indice de Jaccard) nous indique que seulement 9,6% des tokens pour lesquels le système a fait au moins une prédiction a reçu une annotation pour chacune des deux tâches. Ces résultats ne sont pas particulièrement surprenants au vu du processus d'entraînement. Nous remarquons ensuite que l'utilisation de l'auto-apprentissage durant le pré-entraînement améliore grandement la situation, notamment en ce qui concerne l'étiquetage morpho-syntaxique (pour lequel les performances sont alors proches de la borne supérieure). Le renforcement, avec le système de récompense *zéro* et encore plus avec *auto*, s'avère particulièrement efficace, et les meilleures performances sont obtenus en combinant le renforcement avec l'auto-apprentissage.

5 Scénario plus réaliste : glissement de domaine

Let deux premières lignes de la table 2 concernent la configuration COMPLET du corpus mixte GUM+EWT. Les quatre lignes suivantes concerne une configuration similaire, MONO-CORPUS, dans laquelle les modèles sont aussi entraînées sur des données complètes, mais uniquement sur l'un des deux sous-corpus. Les cinq dernières lignes correspondent à la configuration DISJOINT. [4]

3. Les modèles n'étant pas contraints à effectuer des prédictions complètes, les mesures F1 sont naturelles pour chacune des tâches. Notons cependant que plus les taux d'analyse approchent 1, plus les F1 en syntaxe (SYN) et morpho-syntaxe (TAG) correspondent aux mesures habituelles : LAS (*labelled attachement score*) et exactitude.

4. La comparaison entre les configurations MONO-CORPUS est DISJOINT est intéressante mais non évidente. En effet, si les modèles MONO-CORPUS ne sont entraînés que sur l'un des sous-corpus (ce qui les place dans une situation de glissement

Données	Apprentissage	SYN (GUM)	TAG (GUM)	SYN (EWT)	TAG (EWT)	Chev.
GUM+EWT	-AR	0,810(0,810/0,809)	0,956(0,956/0,956)	0,792(0,793/0,791)	0,937(0,937/0,937)	1,000
	+AR[zéro]	0,830(0,841/0,820)	0,960(0,960/0,960)	0,809(0,817/0,800)	0,941(0,941/0,941)	0,994
GUM	-AR	0.783(0.784/0.783)	0.949(0.949/0.949)	**0.719(0.721/0.718)**	**0.904(0.904/0.903)**	1,000
	+AR[zéro]	0,817(0,832/0,802)	0,956(0,956/0,956)	**0,741(0,754/0,728)**	**0,908(0,908/0,908)**	0,994
EWT	-AR	**0.721(0.722/0.720)**	**0.931(0.931/0.931)**	0.766(0.767/0.764)	0.929(0.929/0.929)	1,000
	+AR[zéro]	**0,749(0,768/0,730)**	**0,936(0,936/0,936)**	0,798(0,812/0,785)	0,935(0,935/0,935)	0,987
GUM(SYN) + EWT(TAG)	-AR	0.685(0.730/0.646)	**0.176(0.913/0.108)**	**0.444(0.719/0.328)**	0.632(0.911/0.493)	0,037
	-AR+AA	0.735(0.760/0.712)	**0.938(0.939/0.938)**	**0.634(0.732/0.563)**	0.928(0.928/0.928)	0,936
	+AR[zéro]	0.695(0.757/0.644)	**0.281(0.914/0.177)**	**0.474(0.735/0.356)**	0.711(0.915/0.588)	0,155
	+AR[auto]	0,725(0,730/0,720)	**0,915(0,924/0,907)**	**0,679(0,682/0,675)**	0,903(0,916/0,890)	0,976
	+AR[auto]+AA	0,791(0,833/0,754)	**0.945(0,945/0,945)**	**0,741(0,782/0,705)**	0,934(0,934/0,934)	0,978

TABLE 2 – Performances sur l'UD ; chaque valeur est une moyenne sur 9 exécutions. Format : F1(Précision/Rappel) pour la syntaxe (SYN) et la morpho-syntaxe (TAG). Chev. indique le taux de chevauchement évalué sur la totalité du corpus mixte GUM+EWT. Les valeurs correspondant à des évaluations hors-domaines sont notées **ainsi**.

Nous remarquons ici aussi que le pré-entraînement seul (-AR) mène à des modèles qui n'analysent que très rarement leur entrée à la fois en syntaxe et en morpho-syntaxe. Contrairement à ce qui se passe sur le PTB, cependant, le renforcement avec le système de récompense *zéro* (+AR[zéro]) ne suffit pas à corriger cette faiblesse. L'utilisation du système de récompense *auto* améliore grandement cette situation, sans toutefois mener à des analyses toujours complète. De même, l'auto-apprentissage seul est moins efficace que sur le PTB. L'une des raisons est que notre implémentation actuelle ne force pas l'auto-apprentissage à compléter entièrement les annotations manquantes et que, si cela se produit néanmoins très souvent sur le PTB, cela est moins le cas sur le corpus mixte GUM+EWT (ce que l'on voit notamment en comparant les taux de chevauchement des modèles -AR+AA dans les deux tableaux). Cela explique aussi pourquoi l'utilisation du système de récompense *auto* plutôt que *zéro* (non inclus dans la table) fait une différence même en combinaison avec l'auto-apprentissage.

Les meilleurs résultats, obtenus pour le modèle entraîné en employant la technique d'auto-apprentissage en conjonction avec le système de récompense *auto* lors du renforcement, restent clairement en deçà de celles de la configuration COMPLET. Toutefois, ce modèle obtient des performances en étiquetage morpho-syntaxique en domaine (sur EWT) similaires à celles du modèle MONO-CORPUS entraîné sur EWT et est meilleur que lui hors-domaine (sur GUM). En analyse syntaxique, les résultats sont moins convaincants : le système est moins performant en domaine (sur GUM) que le modèle MONO-CORPUS entraîné sur GUM, et lui est équivalent hors-domaine (sur EWT).

6 Discussion

Nous avons présenté des travaux en cours sur la possibilité d'entraîner des systèmes d'analyse jointe en utilisant des jeux de données disjoints. L'une des méthodes étudiées consiste en l'utilisation d'apprentissage par renforcement avec un système de récompense encourageant le système à effectuer des prédictions même lorsqu'il n'existe pas d'annotations de référence. L'autre méthode, une

de domaine total lorsque évalués sur l'autre sous-corpus), l'utilisation de données de référence complètes lors de l'entraînement est un avantage notable.

forme d'auto-apprentissage, consiste en l'entraînement du modèle sur des données (plus ou moins partiellement) complétées par lui-même. (Nous prévoyons dans un jeu d'expériences plus complet de comparer ces méthodes à l'utilisation d'un système multi-tâche par simple partage de paramètres, construit autour d'un encodeur commun à différents sous-systèmes — un par tâche traitée.)

Sur un corpus homogène, nous avons constaté de nets gains de performances grâce à l'utilisation de ces deux méthodes. Cependant, lorsque les données d'entraînement pour chaque tâche proviennent de corpus distincts, leur efficacité est moindre. Il faut garder en tête les difficultés inhérentes au glissement de domaine (Ramponi & Plank, 2020; Li, 2012), mais nous pouvons aussi pointer du doigt un certain nombre de caractéristiques de l'implémentation actuelle qui interagissent mal avec la configuration DISJOINT. Comme nous travaillons uniquement sur des tâches impliquant un nombre connu de prédictions par token (une dépendance entrante et une étiquette ; contrairement à, par exemple, la tâche d'analyse sémantique traitée par le modèle Bernard 2021 que nous reprenons), il serait possible de forcer le système à effectuer des prédictions complètes. Cela aurait un intérêt en ce qui concerne les sorties effectives du système, mais potentiellement aussi en ce qui concerne la complétion des annotations par auto-apprentissage. À propos de l'auto-apprentissage en particulier, le système récupère, pour chaque phrase, ses propres prédictions concernant l'une des tâches, que nous lui faisons calculer à partir de la suite de tokens brute alors qu'il serait possible de lui fournir aussi les annotations concernant l'autre tâche en entrée. Notons enfin que la fonction de coût minimisée durant le pré-entraînement pénalise non seulement les prédictions fausses, mais aussi celles pour lesquelles il n'existe pas d'annotations. Une alternative — notre prochaine étape — est d'ignorer les probabilités assignées aux actions correspondant à ces dernières, ce qui pourrait avoir un fort impact sur la qualité des modèles pré-entraînés en configuration DISJOINT.

Remerciements

Ces travaux ont bénéficié d'un financement Émergence 2021 (projet SYSNEULING) de l'IdEx Université Paris Cité.

Références

ALBERTI C., WEISS D., COPPOLA G. & PETROV S. (2015). Improved Transition-Based Parsing and Tagging with Neural Networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, p. 1354–1359, Lisbon, Portugal : Association for Computational Linguistics. DOI : 10.18653/v1/D15-1159.

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.

BOHNET B. & NIVRE J. (2012). A Transition-Based System for Joint Part-of-Speech Tagging and Labeled Non-Projective Dependency Parsing. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, p. 1455–1465, Jeju Island, Korea : Association for Computational Linguistics.

CARUANA R. (1997). Multitask Learning. *Machine Learning*, **28**(1), 41–75. DOI : 10.1023/A :1007379606734.

DE MARNEFFE M.-C., MACCARTNEY B. & MANNING C. D. (2006). Generating Typed Dependency Parses from Phrase Structure Parses. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*, Genoa, Italy : European Language Resources Association (ELRA).

DEMPSTER A. P., LAIRD N. M. & RUBIN D. B. (1977). Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, **39**(1), 1–38. 54689.

FINN C., YU T., FU J., ABBEEL P. & LEVINE S. (2017). Generalizing skills with semi-supervised reinforcement learning. In *International Conference on Learning Representations*.

HATORI J., MATSUZAKI T., MIYAO Y. & TSUJII J. (2011). Incremental Joint POS Tagging and Dependency Parsing in Chinese. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, p. 1216–1224, Chiang Mai, Thailand : Asian Federation of Natural Language Processing.

LI Q. (2012). *Literature Survey : Domain Adaptation Algorithms for Natural Language Processing*. Rapport interne, Department of Computer Science, City University of New York.

LI Z., ZHANG M., CHE W., LIU T., CHEN W. & LI H. (2011). Joint Models for Chinese POS Tagging and Dependency Parsing. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, p. 1180–1191, Edinburgh, Scotland, UK. : Association for Computational Linguistics.

MARCUS M. P., SANTORINI B. & MARCINKIEWICZ M. A. (1993). Building a large annotated corpus of English : The Penn Treebank. *Computational Linguistics*, **19**(2), 313–330.

NG A. Y. & RUSSELL S. J. (2000). Algorithms for inverse reinforcement learning. In *Proceedings of the Seventeenth International Conference on Machine Learning*, ICML '00, p. 663–670, San Francisco, CA, USA : Morgan Kaufmann Publishers Inc.

NGUYEN D. Q. & VERSPOOR K. (2018). An Improved Neural Network Model for Joint POS Tagging and Dependency Parsing. In *Proceedings of the CoNLL 2018 Shared Task : Multilingual Parsing from Raw Text to Universal Dependencies*, p. 81–91, Brussels, Belgium : Association for Computational Linguistics. DOI : 10.18653/v1/K18-2008.

NIGAM K. & GHANI R. (2000). Analyzing the effectiveness and applicability of co-training. In *Proceedings of the ninth international conference on Information and knowledge management*, CIKM '00, p. 86–93, New York, NY, USA : Association for Computing Machinery. DOI : 10.1145/354756.354805.

NIVRE J., DE MARNEFFE M.-C., GINTER F., HAJIČ J., MANNING C. D., PYYSALO S., SCHUSTER S., TYERS F. & ZEMAN D. (2020). Universal Dependencies v2 : An evergrowing multilingual treebank collection. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, p. 4034–4043, Marseille, France : European Language Resources Association.

OEPEN S., KUHLMANN M., MIYAO Y., ZEMAN D., CINKOVA S., FLICKINGER D., HAJIC J. & URESOVA 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 : Association for Computational Linguistics. DOI : 10.18653/v1/S15-2153.

PENG H., THOMSON S., SWAYAMDIPTA S. & SMITH N. A. (2018). Learning Joint Semantic Parsers from Disjoint Data. In *Proceedings of the 2018 Conference of the North American Chapter*

of the Association for Computational Linguistics : Human Language Technologies, Volume 1 (Long Papers), p. 1492–1502, New Orleans, Louisiana : Association for Computational Linguistics. DOI : 10.18653/v1/N18-1135.

PENNINGTON J., SOCHER R. & MANNING C. D. (2014). GloVe : Global Vectors for Word Representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, p. 1532–1543.

RAMPONI A. & PLANK B. (2020). Neural unsupervised domain adaptation in NLP—A survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, p. 6838–6855, Barcelona, Spain (Online) : International Committee on Computational Linguistics. DOI : 10.18653/v1/2020.coling-main.603.

SILVEIRA N., DOZAT T., DE MARNEFFE M.-C., BOWMAN S., CONNOR M., BAUER J. & MANNING C. (2014). A gold standard dependency corpus for English. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, p. 2897–2904, Reykjavik, Iceland : European Language Resources Association (ELRA).

SUTTON R. S. & BARTO A. G. (2018). *Reinforcement Learning : An Introduction*. Adaptive Computation and Machine Learning series. MIT Press, second edition édition.

WILLIAMS R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, **8**(3-4), 229–256. DOI : 10.1007/BF00992696.

ZELDES A. (2017). The GUM corpus : Creating multilayer resources in the classroom. *Language Resources and Evaluation*, **51**(3), 581–612. DOI : http ://dx.doi.org/10.1007/s10579-016-9343-x.

ZHANG Y. & YANG Q. (2017). A survey on multi-task learning. DOI : 10.48550/ARXIV.1707.08114.

ZHAO F. (2022). Auto-correction dans un analyseur neuronal par transitions : un comportement factice ? In *Actes de la 29e conférence sur le Traitement Automatique des Langues Naturelles et des 24es Rencontres des Etudiants Chercheurs en Informatique et Traitement Automatique des Langues*, volume RECITAL, p. 20–32, Avignon, France : ATALA.