



# **Génération automatique d'exercices de syntaxe à partir de corpus annotés.**

**Mémoire présenté par**

**Thuy Thi PHAM**  
Université de Lille

Mémoire de M2 mention SDL parcours LTTAC

## **JURY**

**Antonio Balvet**, Maître de conférences et Enseignant Chercheur en Sciences du Langage à l'Université de Lille. (directeur)

**Katia Paykin Arroues**, Maîtresse de conférences et Enseignant Chercheur en Sciences du Langage à l'Université de Lille. (examinatrice)

**Lille - 2023**

## **REMERCIEMENTS**

J'aimerais tout d'abord exprimer ma sincère gratitude envers mon directeur de mémoire, Antonio Balvet, professeur au département de Science du langage de l'université de Lille. Sa patience, sa disponibilité et ses conseils éclairés ont vraiment aidé à nourrir ma réflexion tout au long de mon parcours.

Je remercie également toute l'équipe pédagogique de l'université de Lille ainsi que les intervenants professionnels qui ont contribué à la partie théorique de ma formation. Votre engagement a joué un rôle clé dans mon acquisition de connaissances.

J'adresse ma reconnaissance la plus profonde aux membres du jury, dont l'acceptation de lire et d'évaluer ce mémoire a été une étape cruciale dans ce processus académique.

Enfin, je n'oublie pas ma famille. Je leur suis vraiment reconnaissante pour leur amour, pour m'avoir toujours soutenu et encouragé même si nous sommes loin les uns des autres.

Merci, et, je le souhaite, bonne lecture.

## Table des matières

REMERCIEMENTS.....	2
I - INTRODUCTION.....	5
II – ÉTAT DE L’ART DES PLATEFORMES EN LIGNE.....	9
1. H5P.....	9
1.1 Présentation générale H5P.....	9
1.2. Exercices interactifs.....	10
1.2.1. Questionnaire à Choix Multiples (QCM).....	10
1.2.2. « Mark the words » (Sélectionner les bons mots) .....	11
1.2.3 Find Multiple Hotspots.....	12
1.2.4. Drag and Drop (Glisser-déposer).....	14
1.3. Inconvénients de H5P.....	15
2. Moodle.....	17
2.1 Présentation générale Moodle.....	17
2.2 Avantages et fonctionnalités spécifiques de Moodle.....	17
2.3 Inconvénients de Moodle.....	19
3. WIMS.....	20
3.1 Présentation générale WIMS.....	20
3.2 Fonctions des exercices de WIMS.....	20
3.3 Exercices de syntaxe sur WIMS.....	21
3.4 Expérience personnelle de réalisation d’exercices sur WIMS.....	23
3.5 Commentaires sur les consignes des exercices de syntaxe.....	26
3.5.1 Différents types de phrases.....	26
3.5.2 Exercices théoriques.....	27
3.6 Conclusion concernant la partie de l’application WIMS.....	28
4. Conclusion concernant l’état de l’art sur H5P, MOODLE et WIMS.....	29
III - DÉPENDANCE SYNTAXIQUE.....	29
1. Qu’est-ce que la dépendance syntaxique ?.....	29
2. Universal Dependencies (UD).....	33
2.1 Format CoNLL-U.....	33
2.2 Générer des graphiques.....	35
IV – ÉVALUATION DES CORPUS AVEC UDPIPE, SPACY, STANZA.....	35
1. Introduction.....	35
2. Présentation générale UDPipe.....	36
3. Présentation générale Spacy.....	37
4. Présentation générale Stanza.....	37
5. Évaluation du corpus contenant un attribut du COD.....	38
Conclusion du corpus contenant un attribut du COD.....	46
6. Évaluation du corpus contenant un complément du COD ou un complément du COI.....	47
Conclusion pour l’évaluation de corpus contenant un complément du COD/COI.....	58
7. Conclusion de l’évaluation des corpus avec UDPIPE, Spacy, Stanza.....	59
V – RÉALISATION TECHNIQUE.....	60
1. Présentation globale de projet.....	60
2. Choix du type d’exercice.....	60

3. Choix de l'exercice.....	61
3.1 Choix de l'exercice sur la partie du discours.....	61
3.2 Choix de l'exercice de la fonction grammaticale : COD et COI.....	62
4. Extraction des données pour alimenter notre application.....	63
4.1 Choix technique.....	63
4.2 Préparation des données pour l'exercice Partie du discours (fichier QuizPartieDeDiscours).....	63
.....	65
5. Arbre syntaxique.....	65
6. Préparation des données pour l'exercice sur la fonction grammaticale (fichier QuizCODCOI).....	66
7. Correction pour la partie fonction grammaticale (fichier CorrectionCOD_COI).....	69
8. Choix du format JSON.....	71
9. Création de l'application Web.....	72
9.1 Choix techniques.....	72
9.2 Les fonctions essentielles de l'application de quiz.....	72
9.2.1 L'ordre aléatoire des questions.....	72
9.2.2 Le compte à rebours.....	73
9.2.3 Limiter le nombre de tentatives.....	73
9.2.4 Barre de progression.....	74
9.2.5 Choix de bouton.....	76
9.3 Le graphique présentant les réponses des étudiants.....	76
9.4 L'accessibilité d'un site WEB.....	79
9.4.1 La police du texte.....	81
9.4.2 Taille du texte.....	81
9.4.3 Alignement.....	82
9.4.4 L'interligne.....	82
VI – AXES D'AMÉLIORATION DU PROJET.....	83
1. Point technique.....	83
2. Point de linguistique.....	85
3. Créer d'autres activités que des Quiz.....	85
4. Accessibilité de notre application.....	85
VII - CONCLUSION.....	86
BIBLIOGRAPHIE.....	88

## I - INTRODUCTION

Il est fréquent que les étudiants en première et deuxième année rencontrent des difficultés à mettre en pratique les méthodes linguistiques, telles que l'analyse syntaxique, phonétique et phonologique, ou encore l'analyse de la construction des mots. Quelles sont alors les raisons de ces difficultés ?

Tout d'abord, il est important de noter que l'enseignement de la grammaire à l'école (de la primaire au lycée) et celui dispensé à l'université, poursuivent des objectifs différents. En effet, « *l'enseignement de la grammaire à l'école doit rester au service de l'expression et de la compréhension* »<sup>1</sup> (Caroline Lachet). En revanche, dans les cours universitaires de linguistique, l'objectif est de développer des compétences analytiques pour comprendre comment les êtres humains utilisent le langage. L'étudiant doit donc être en mesure d'identifier et d'analyser les unités minimales du langage sur les plans phonétique, phonologique, morphologique, syntaxique et sémantique. C'est pourquoi les étudiants en licence ne possèdent pas nécessairement de solides compétences en analyse linguistique lorsqu'ils entrent à l'université.

Caroline Lachet a mené une enquête sur les questions grammaticales posées par les étudiants de licence 1, licence 2 et licence 3 à l'université d'Artois. Voici quelques exemples de ces questions :

- « 5. Dans la phrase suivante *L'enfant court autour de la table*,  
soulignez les noms. Expliquez comment vous les avez repérés.
6. Dans la phrase suivante *L'enfant court autour de la table*,  
soulignez le verbe. Expliquez comment vous l'avez repéré.
7. Dans la phrase suivante *Depuis le début de l'année, Pierre bat Paul*,  
soulignez le sujet. Expliquez comment vous l'avez repéré.
8. Dans la phrase suivante *Depuis le début de l'année, Pierre bat Paul*,  
soulignez le COD. Expliquez comment vous l'avez repéré. »

---

1 *Pratiques et représentations grammaticales des étudiants à l'université*, Caroline Lachet, p.78

Selon cette enquête, il ressort que les étudiants parviennent aisément à identifier le nom, le verbe et le sujet. En revanche, ils rencontrent des difficultés dans l'identification de la fonction objet, avec seulement 68 % de bonnes réponses. De plus, 15 % des étudiants, principalement des L1 et des L2, confondent l'objet avec la fonction circonstancielle. Par ailleurs, la majorité des étudiants de L1 (35 % des L1, 3 % des L2 et 5 % des L3) n'expliquent pas comment reconnaître le nom. Certains étudiants de L1 identifient automatiquement le nom sans recourir à des critères morphologiques, sémantiques ou syntaxiques.

En revanche, les étudiants de L2 et L3 expliquent leurs réponses en utilisant des critères morphologiques, contextuels et syntaxiques. Par exemple, certains déclarent : *"Le verbe est court car ici c'est l'action. De plus, un verbe peut être conjugué"* ou *"Je l'ai trouvé car il est placé derrière le nom 'enfant' qui est le sujet de la phrase"* ou encore *"J'ai repéré le verbe car le COD est un complément du verbe et je me suis posé la question 'Pierre bat qui ?': Paul : COD"*.

Les résultats de cette enquête montrent donc que les étudiants de première année présentent de grosses lacunes pour tout ce qui concerne l'analyse linguistique.

Par ailleurs, en France, l'entrée à l'université, notamment dans le département de Sciences du langage, ne nécessite pas le passage d'un examen pour évaluer le niveau linguistique. Les universités accueillent tout étudiant titulaire du baccalauréat, ce qui peut être à l'origine de disparités dans le niveau de compétence en analyse linguistique des étudiants.

Un exemple concret est donné par les enseignants du département Sciences du langage à l'Université de Lille. Ils constatent que lorsque les étudiants ne connaissent pas un mot dans une phrase, ils se bloquent et ont des difficultés à appliquer la méthode d'analyse structurale.

Ceci amène à se poser une question essentielle : comment pouvons-nous aider les étudiants à renforcer les compétences linguistiques et à faciliter leur intégration dans les cours de linguistique à l'université? D'un côté, les enseignants sont confrontés à une contrainte de temps : ils doivent à la fois corriger les exercices de centaines d'étudiants et créer des activités pédagogiques. D'autre part, les étudiants ne disposent pas forcément des ressources et exercices leur permettant de travailler et progresser de manière autonome.

Plusieurs logiciels existent déjà pour créer des exercices de manière simple et intuitive (comme H5P, Learning App, WIMS, etc.). Cependant, aucun de ces logiciels ne permet la génération automatique à partir de textes ou de ressources linguistiques.

Le projet ACE, dirigé par Monsieur Balvet, enseignant - chercheur en linguistique appliquée à l'université de Lille, a ouvert la voie à la création automatique de questions à choix multiples (QCM) pour Moodle. ACE utilise des corpus annotés générés à l'aide d'Universal Dependencies (UD), un projet coopératif qui vise notamment à élaborer un schéma d'annotation syntaxique et ce dans plus de 100 langues différentes. Cependant, le métalangage de UD ne peut pas être utilisé tel quel pour créer des exercices de manière automatique. Ceci est vrai notamment pour les parties du discours (par exemple : ADP = préposition, alors que dans les corpus anglais, c'est PRP). Si nous utilisons directement le métalangage de UD, les étudiants risquent de ne pas le comprendre, d'où la nécessité d'adapter le métalangage linguistique en un langage compréhensible par les étudiants, tel que préposition, complément d'objet direct, complément d'objet indirect, attribut du COD, COI, etc. Nous allons étudier dans notre projet la possibilité d'automatiser ces « transformations » en un format compréhensible pour le public français.

Comme mentionné précédemment, les exercices générés automatiquement par le projet ACE sont spécifiquement adaptés à l'intégration dans la plateforme Moodle. Or, l'utilisation de Moodle présente quelques inconvénients, ou du moins quelques limitations. Tout d'abord, seuls les étudiants inscrits à des cours sur Moodle, tels que le cours de syntaxe par exemple, sont en mesure de réaliser ces exercices. Il serait intéressant que les futurs étudiants puissent, avant même leur rentrée en licence science du langage, avoir accès à des ressources leur permettant de travailler sur leurs lacunes, notamment en ce qui concerne l'analyse syntaxique.

De plus, les utilisateurs de Moodle (qu'ils soient étudiants ou enseignants) ne peuvent personnaliser de manière individuelle et en profondeur l'affichage des exercices. Cela peut poser problème notamment pour des publics ayant des besoins spécifiques tels que les malvoyants, les personnes atteintes de dysorthographe ou encore de dyspraxie. Il est en effet impossible de choisir la taille de la police, la couleur du texte, la couleur de fond etc ...).

Enfin, Moodle ne nous permet pas d'enregistrer de façon ciblée le temps de réponse de chaque utilisateur pour chaque question. En effet, il nous paraît très intéressant de pouvoir

savoir (à la fois pour l'étudiant ou pour le professeur), sur quelle question l'étudiant a hésité, qu'avait-il initialement répondu avant de modifier sa réponse etc ...

Pour répondre aux problèmes mentionnés ci-dessus, nous nous sommes donc posé les questions suivantes: Comment peut-on s'affranchir de Moodle ? En utilisant à la fois les compétences linguistiques et informatiques que nous avons acquises lors de notre Master, est-il possible de créer notre propre application permettant au plus grand nombre d'utilisateurs possible d'effectuer les exercices que nous avons créé et préparé automatiquement ?

Notre projet va donc s'orienter sur deux axes principaux :

- L'annotation de corpus et la création automatisée de quiz à partir d'un corpus donné
- La création d'une application WEB permettant aux utilisateurs d'effectuer des exercices créés à partir des quiz générés automatiquement

Dans la première partie de notre mémoire nous parlerons de l'état de l'art en ce qui concerne les plateformes de création et de pratique d'exercices. Nous nous concentrerons principalement sur les plateformes H5P, Moodle et WIMS, et nous pencherons également sur les différents types d'exercices sur lesquels nous pourrions travailler (QCM, mark the word, glisser-déposer ...)

Dans la seconde partie de notre mémoire, nous parlerons de la dépendance syntaxique, et présenterons l'un de ses projets piliers, Universal dependencies. Ensuite, nous parlerons du format le plus utilisé pour les structures de dépendances, le format CoNLL-U (lui même une extension de CoNLL)

Dans la troisième partie, nous évaluerons les différents outils d'analyse syntaxique (Spacy, Stanza et UDPipe notamment), et comparerons les résultats obtenus avec chacun de ces outils, dans le but de choisir à la fois l'outil qui convient le mieux pour notre projet, mais aussi le ou les corpus que nous utiliserons pour générer nos exercices.

La dernière partie portera sur la réalisation technique du projet, d'une part sur la préparation des données (de l'annotation des corpus jusqu'à l'élaboration d'un questionnaire), et d'autre part sur la création de notre application web, et l'intégration des questionnaires que nous aurons générés.



Enfin, nous parlerons des différentes améliorations/nouveautés qui pourraient être apportées à notre projet et finirons par une conclusion.

## **II – ÉTAT DE L'ART DES PLATEFORMES EN LIGNE**

### **1. H5P**

#### **1.1 Présentation générale H5P**

H5P (HTML5 Package) est un outil qui permet de créer une grande diversité de contenus interactifs, basé sur HTML5 et Javascript. Il est développé depuis 2014.

H5P présente les avantages d'être gratuit, open source et d'avoir une interface utilisateur facile à prendre en main. De plus, les contenus interactifs créés peuvent être utilisés aussi bien sur ordinateur que sur une tablette ou un smartphone.

H5P est facile à utiliser et ne demande aucune compétence en programmation. Il suffit de sélectionner le type de contenu que l'on souhaite créer et de remplir les champs requis. L'interface intuitive facilite le processus de création, permettant ainsi aux enseignants de créer du contenu interactif HTML5 sans difficulté.

Une fois le contenu créé, il peut être partagé et réutilisé facilement via un navigateur web. Cette fonctionnalité permet de diffuser et de distribuer rapidement des ressources d'apprentissage interactives à un large public.

De plus, H5P est disponible en tant que plugin pour les systèmes de publication populaires tels que Moodle, WordPress, Drupal, et d'autres. Cela offre aux programmeurs et aux professionnels de l'auto-hébergement une intégration aisée avec leurs plateformes existantes, permettant une utilisation fluide et une personnalisation avancée.

Il convient de noter que H5P peut également être utilisé hors ligne sur un ordinateur en utilisant les logiciels Logiquiz ou Lumi. Cette option permet une utilisation pratique même en l'absence de connexion Internet, offrant une flexibilité supplémentaire pour créer et accéder aux exercices.

En résumé, H5P offre une expérience utilisateur accessible et pratique, permettant à chacun de créer du contenu interactif HTML5 sans compétences techniques approfondies. Sa

compatibilité avec différentes plateformes de publication en fait un outil polyvalent pour divers contextes d'apprentissage et de diffusion de contenu interactif.

## 1.2. Exercices interactifs

H5P offre la possibilité de créer dans Moodle des ressources d'apprentissage interactives au format HTML5, combinant du texte, des images, des vidéos, des présentations et des quiz. Ces ressources d'apprentissage sont accessibles à tout moment, que ce soit sur mobile ou sur ordinateur, ce qui permet aux étudiants d'étudier et de s'auto-évaluer à leur propre rythme et de manière autonome (Rekhari & Sinnayah, 2018).

### 1.2.1. Questionnaire à Choix Multiples (QCM)

Dans H5P, l'une des activités disponibles est le QCM. Cette activité permet de tester le niveau de connaissance de l'apprenant sur un sujet donné. Les questions à choix multiples dans H5P peuvent comporter une ou plusieurs options correctes par question.

#### a) Types de questions

H5P donne à l'utilisateur la possibilité de choisir parmi différents types de questions : automatique, choix multiple ou choix unique. Dans le cas du choix multiple, l'utilisateur peut sélectionner plusieurs options correctes, et les cases à cocher correspondantes seront affichées. Pour le choix unique, une seule option correcte est attendue, et les boutons radio correspondants seront affichés. Avec différents types de questions possibles, l'utilisateur peut personnaliser les quiz comme il le souhaite.

You will have something like this in the editor:

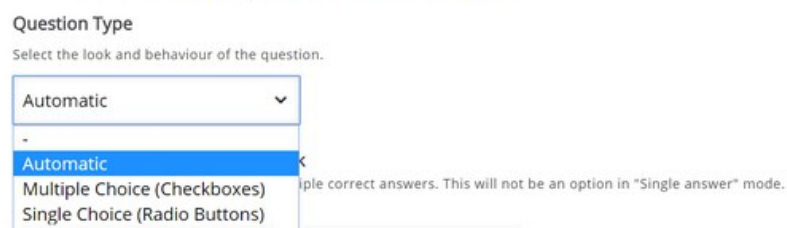


Figure 1: Sélectionner les types des questions

## b) Ordre aléatoire

De plus, l'utilisateur a la possibilité d'afficher les réponses dans un ordre aléatoire en cochant la case « Randomize answers » (Mélanger les réponses). L'option « Randomize answers » contribue à diversifier l'expérience d'apprentissage et à renforcer la fiabilité de l'évaluation.

## c) Réessayer le test

Afin de permettre à l'apprenant de corriger ses erreurs et de renforcer sa compréhension, H5P propose le bouton « Réessayer » (« Try again ») pour refaire le test.

### 1.2.2. « Mark the words » (Sélectionner les bons mots)

« Mark the Words » est un jeu interactif sur la plateforme d'apprentissage H5P. Dans ce jeu, on demande à l'apprenant de sélectionner des mots en fonction de la consigne de l'exercice. Cette activité favorise la compréhension et la mémorisation, ce qui contribue à la concentration des élèves.

Pour faire cet exercice, nous avons utilisé une application appelée Lumi. Dans Lumi, nous avons choisi l'option « Mark the Word ». Ensuite, nous avons écrit les consignes et le texte de l'exercice dans les espaces prévus. Dans la zone où se trouve le texte, nous avons tapé le contenu et mis des astérisques (\*) autour des mots sur lesquels nous voulions que les apprenants cliquent. Voici un exemple d'exercice que nous avons créé :

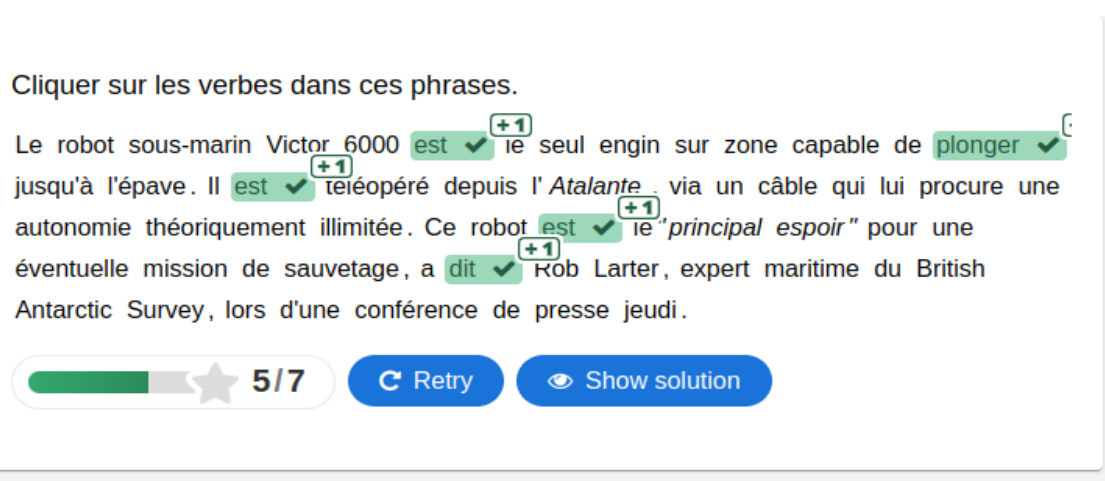


Figure 2: Exercice "Click on the verbs" réalisé avec le modèle "Mark the Word" de H5P à l'aide de l'application Lumi.

La création de cet exercice se fait manuellement, ce qui signifie que les enseignants doivent taper ou copier un texte depuis Internet. Ensuite, pour corriger, les enseignants doivent le faire eux-mêmes et vérifier plusieurs fois si les bons mots sont bien choisis. Tout ça prend beaucoup de temps, surtout si l'on oublie de mettre des astérisques autour des mots.

### **1.2.3 Find Multiple Hotspots**

H5P propose également l'activité « Find Multiple Hotspots ». Dans cette activité, les apprenants sont présentés avec une image et sont invités à trouver et cliquer sur plusieurs zones spécifiques à l'intérieur de l'image. Ce format de contenu peut être utilisé avec H5P dans divers systèmes de publication tels que Canvas, Brightspace, Blackboard, Moodle et WordPress.<sup>2</sup>

Ce type d'exercice présente plusieurs avantages en terme d'accessibilité et d'engagement. Il est facile à utiliser, que ce soit sur un ordinateur, une tablette ou même un téléphone. L'utilisateur n'a qu'à cliquer sur un endroit de l'image, et s'il s'agit d'une bonne réponse, il le verra à l'aide d'un « check » affiché en vert. De plus, la présentation visuelle de cet exercice est attrayante et bien réalisée.

Les exercices de type « Find Multiple Hotspots » sont particulièrement utiles pour des activités impliquant des objets concrets tels que les parties du corps, les légumes, les fruits etc.

---

2 <https://h5p.org/find-multiple-hotspots>

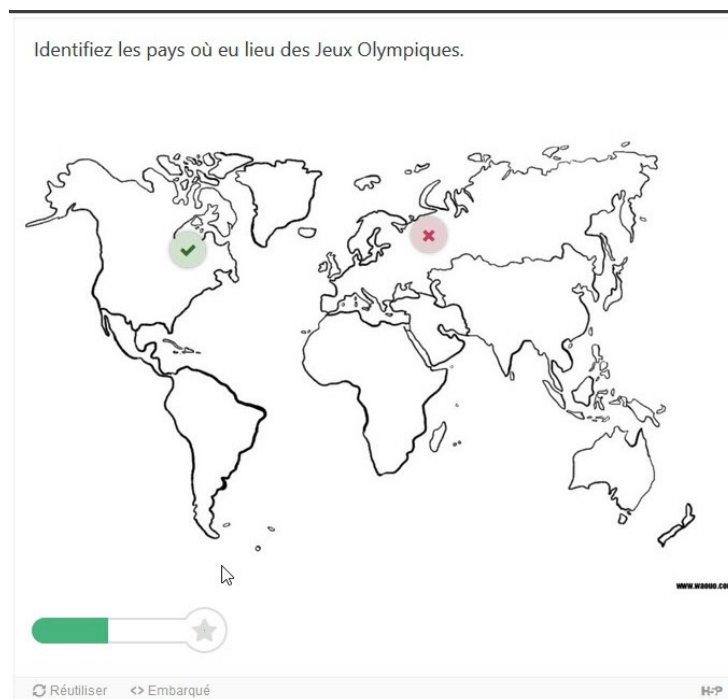


Figure 3: Contenus interactifs sur H5P (Université de Sherbrooke), consulté le 19/06/2023

Find all the vegetables in this picture

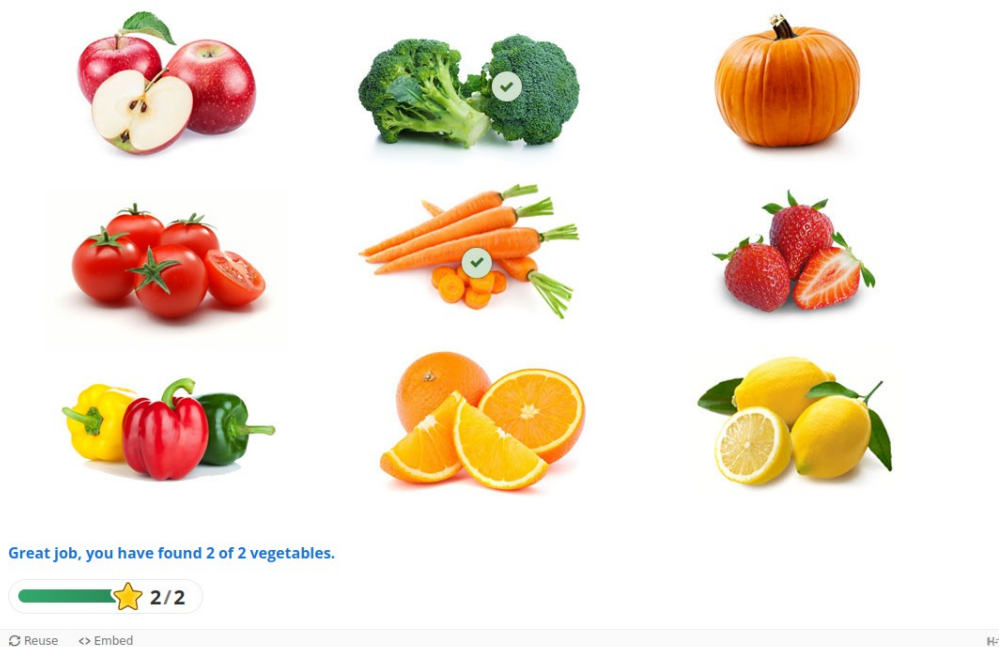


Figure 4: Contenu interactif sur le site H5P, consulté le 19/06/2023

Lorsqu'il s'agit de créer des exercices de syntaxe, ce type d'exercice n'est vraiment pas idéal. Tout d'abord, il nécessite une création manuelle des exercices à partir d'un texte, suivi d'une conversion en image. De plus, notre objectif est de générer des exercices automatiquement, ce qui n'est pas possible avec ce type d'exercice, car il ne permet pas une détection automatique des bonnes réponses. Par conséquent, cette activité n'est pas adaptée pour les exercices de syntaxe que nous souhaitons créer.

#### 1.2.4. Drag and Drop (Glisser-déposer)

H5P propose aussi des outils pour créer des activités « glisser-déposer ». Dans ce jeu, les participants doivent déplacer des éléments d'une position à une autre sur l'écran à l'aide de la souris ou du toucher, en fonction d'un objectif ou d'une règle spécifique.

Le glisser-déposer est utilisé pour regrouper des éléments partageant des caractéristiques communes, associer un objet à un autre, organiser des éléments dans le bon ordre, etc.

Nous avons développé le jeu de glisser-déposer axé sur la pratique de l'utilisation des conjonctions de subordination et de coordination en les plaçant correctement. Voici le jeu de glisser-déposer que l'on a réalisé :

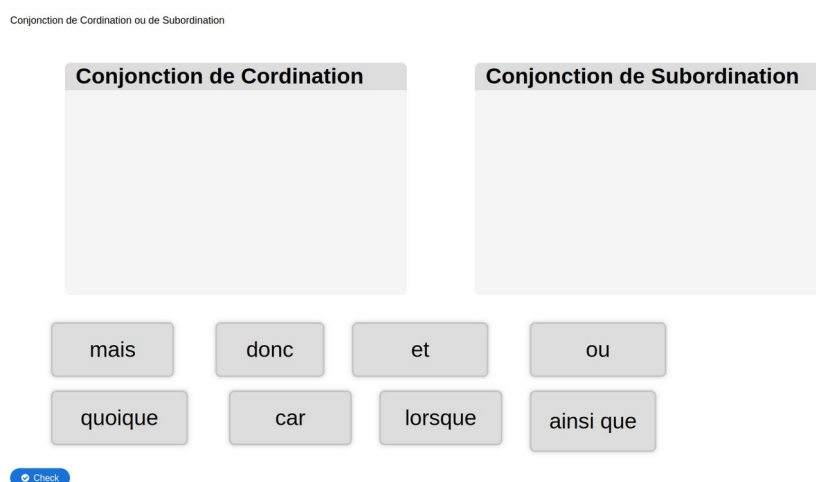


Figure 5: Jeu glisser-déposer H5P, fait le 20/06/2023

Ce jeu a été entièrement conçu manuellement. Nous avons commencé par créer les zones de dépôt où les éléments doivent être placés. Ensuite, nous avons élaboré les différentes réponses disponibles en sélectionnant soigneusement les bonnes réponses pour chaque zone de dépôt. Cependant, ce processus de création manuelle présente un inconvénient majeur en termes de

temps requis. Par exemple, la rédaction de 100 réponses peut prendre au moins une heure. De plus, lorsqu'on ajoute de nouvelles réponses, il est nécessaire de redimensionner les zones de dépôt afin d'éviter tout problème d'affichage des résultats.

### 1.3. Inconvénients de H5P

#### Suivi des étudiants

Lorsque H5P est utilisé seul, il ne permet pas de suivre la progression des élèves. Cependant, si les exercices sont intégrés dans Moodle, on peut observer le progrès des étudiants. De plus, pour les activités qui nécessitent une évaluation, les résultats sont enregistrés dans le carnet de notes de Moodle.<sup>3</sup>

On peut voir ci-dessous un exemple d'exercice « **Mark the words** », créé par Monsieur Balvet, enseignant-chercheur en linguistique à l'université de Lille. Ces exercices sont intégrés dans Moodle afin que les étudiants en Licence 1 de la Science du langage puissent améliorer les méthodes d'analyse dans la morphologie, la sémantique, la syntaxe, etc. La consigne ici est de marquer tous les mots grammaticaux du texte « *L'argent* » de E.Zola.

Tutorats licence 1 SDL

Accueil > Mes cours > Faculté des Humanités > Sciences du Langage (SDL) > Licence SDL > Tutorats L1 SDL > Syntaxe > Exercice: trouver les mots grammaticaux

Exercice: trouver les mots grammaticaux

Consulter mes tentatives

Marquer tous les mots grammaticaux de ce texte de E. Zola (*L'Argent*).

Petit, brun, très vir, joli homme, il venait d'hériter de la charge d'un neveu ses oncles, à trente-deux ans. Et il semblait tout au convive qu'il avait en face de lui un gros monsieur à figure rouge et rasée, le célèbre Amadieu, que la Hourse vénérait, depuis son fameux coup sur les Mines de Selsis. Lorsque les titres étaient tombés à quinze francs, et que l'on considérait tout acheteur comme un fou, il avait mis dans l'affaire sa fortune, deux cent mille francs, au hasard sans calcul ni flair, par un entêtement de brute chanceuse. Aujourd'hui que la découverte de mines réels et considérables avait fait dépasser aux titres le cours de mille francs, il gagnait une quinzaine de millions; et son opération imbécile qui aurait dû le faire enfermer autrefois, le haussait maintenant au rang des vastes cerveaux financiers. Il était salué, consulté surtout d'ailleurs, il ne donnait plus d'ordres, comme satisfait, trônant désormais dans son coup de génie unique et légendaire. Mazaud devait rêver sa clientèle.

A

74/91

Recommencer

Voir la correction



Figure 6: Exercice réalisé avec "Mark the Words" H5P sur la plateforme Moodle de l'Université de Lille.

Lorsque l'on clique en haut à droite sur « Consulter mes tentatives », on peut visualiser plusieurs informations concernant notre participation à cet exercice. Voici le tableau de mes tentatives :

<sup>3</sup> <https://warwick.ac.uk/services/academictchnology/support/guides/moodle-guides/mdl-82/>

## TP Mes tentatives

### Tentative avec le score le plus élevé

#	Date	Score	Score max	Durée	Complétude	Succès	Rapport
9	22 juin 2023, 11:33	74	91	1 minutes 47 secondes			<a href="#">Consulter le rapport</a>

### Toutes les tentatives utilisateur








#	Date	Score	Score max	Durée	Complétude	Succès	Rapport
1	27 février 2023, 22:06	0	91	30 secondes			<a href="#">Consulter le rapport</a>
2	27 février 2023, 22:07	4	91	1 minutes 16 secondes			<a href="#">Consulter le rapport</a>
3	1 mars 2023, 13:33	27	91	1 minutes 35 secondes			<a href="#">Consulter le rapport</a>
4	2 mars 2023, 12:28	23	91	35 minutes 50 secondes			<a href="#">Consulter le rapport</a>
5	7 mars 2023, 16:37	2	91	7 secondes			<a href="#">Consulter le rapport</a>

Figure 7: Tableau de mes tentatives sur Moodle

Ce tableau présente un aperçu de mes tentatives sur la plateforme Moodle. Chaque ligne correspond à une tentative et fournit des informations telles que la date de la tentative, le nombre total de tentatives effectuées, la note obtenue, la note maximale possible, la durée de la tentative, et si l'exercice a été complété avec succès. De plus, nous pouvons également consulter tous les mots que nous avons déjà sélectionnés. Grâce à l'utilisation de H5P dans Moodle, il est possible d'accéder à l'évaluation de chaque étudiant. Par exemple, dans le cas présent, j'ai effectué cet exercice à 9 reprises. En observant les scores et les durées, je constate que j'ai progressé à chaque tentative (première fois : score de 0 en 30 secondes, dernière fois : score de 74 en 1 minute 47 secondes).

Malgré tout, on peut regretter l'absence de certaines fonctionnalités :

#### - L'impossibilité de créer des exercices de manière automatique.

Lors de la création des activités avec l'outil H5P, l'utilisateur est contraint de les créer manuellement, ce qui demande beaucoup de temps. Par exemple, si l'on souhaite créer 100 questions, cela nécessite au moins 2 heures de travail. Il est donc primordial de trouver une solution qui permette de générer automatiquement des questions à choix multiples, afin d'économiser considérablement le temps des enseignants.



### **- L'impossibilité d'insérer un arbre syntaxique**

Dans H5P il n'est pas possible d'insérer un arbre syntaxique pour permettre aux étudiants de visualiser les relations entre les mots dans une phrase.

### **- L'absence de fonctionnalités d'accessibilité.**

H5P est un outil pratique pour créer des QCM, *Mark the words*, Glisser-déposer. Cependant, il ne propose pas de fonctionnalités d'accessibilité. Il n'est pas possible d'apporter des modifications, telles que changer la couleur des boutons, ajuster la taille ou la police du texte, etc.. Cette limitation est problématique, car elle restreint la capacité des personnes ayant des limitations visuelles ou cognitives à naviguer efficacement sur le contenu.

## **2. Moodle**

### **2.1 Présentation générale Moodle**

Moodle est une plateforme d'apprentissage en ligne qui permet la création d'un espace d'apprentissage virtuel. Elle est également connue sous le nom de LMS (Learning Management System)<sup>4</sup>. Créé par Martin Dougiamas en 2002<sup>5</sup>, Moodle est un logiciel open source distribué sous la licence GPL (General Public License)<sup>6</sup>, ce qui signifie que son contenu peut être modifié et adapté selon les besoins des utilisateurs. À ce jour, Moodle compte plus de 368 millions d'utilisateurs et 151 381 sites enregistrés dans le monde entier. Le site est présent dans 239 pays.<sup>7</sup>

### **2.2 Avantages et fonctionnalités spécifiques de Moodle**

- Les activités<sup>8</sup> : Elles regroupent toutes les fonctionnalités que l'on peut utiliser au sein d'un cours Moodle. Les activités permettent aux étudiants d'interagir avec les enseignants ou d'autres étudiants. Elles incluent des éléments tels que les devoirs, les chats, les forums, les leçons, les quiz, les wikis, etc.

---

4 Learning management system, juin 2023, [https://fr.wikipedia.org/wiki/Learning\\_management\\_system](https://fr.wikipedia.org/wiki/Learning_management_system)

5 The Moodle Story, juin 2023, <https://moodle.com/about/the-moodle-story/>

6 Licence publique générale gnu, juin 2023, <http://www.gnu.org/copyleft/gpl.html>.

7 Moodle statistics, juin 2023, <https://moodle.net/stats/>.

8 Activities, juin 2023, [https://docs.moodle.org/31/en/Lesson\\_activity](https://docs.moodle.org/31/en/Lesson_activity)

- Les ressources<sup>9</sup>: Ce sont des éléments créés par les enseignants pour soutenir l'apprentissage des étudiants. Les ressources sont destinées à la consultation uniquement et comprennent des éléments tels que les livres, les fichiers, les dossiers, les pages web, etc.
- Les blocs<sup>10</sup> : Les blocs fournissent une vue agrégée des ressources ou des activités. Les administrateurs peuvent installer de nouveaux blocs que les enseignants peuvent activer ou désactiver dans leurs cours. Ces blocs peuvent être placés à différents endroits sur la page et peuvent apparaître à différents niveaux, tels que sur la page d'accueil ou dans un cours. Moodle propose également des blocs standards tels que le calendrier, les commentaires, le résumé du cours, les messages, la navigation, etc.
- Les rapports<sup>11</sup> : Les rapports sont des pages sur le site où l'on peut obtenir des informations spécifiques sur certains aspects. À l'origine, il existait des rapports de cours et des rapports d'administration, mais ces deux types de rapports ont été fusionnés pour ne former qu'une seule catégorie.
- Le carnet de notes<sup>12</sup> : Le carnet de notes est un outil qui permet de rassembler toutes les notes obtenues par les étudiants dans un cours. Chaque fois qu'un étudiant effectue une activité qui sera évaluée, la note est automatiquement ajoutée au carnet. Il permet aux étudiants et aux enseignants de consulter les notes, aux enseignants de les modifier, de les regrouper en catégories, etc. En résumé, c'est l'endroit où l'utilisateur peut trouver des informations sur ses notes. Le carnet de notes permet également de récupérer la note totale d'un étudiant pour un cours donné.
- Le Suivi des réalisations<sup>13</sup> : Le suivi des réalisations permet aux enseignants de définir des activités qui doivent être complétées pour réussir le cours sur la plateforme. Cela permet aux étudiants de visualiser leur progression dans le cours et de savoir ce qu'il leur reste à accomplir pour avancer. C'est un outil précieux pour le suivi des étudiants, mais il doit être activé manuellement par l'enseignant dans le menu d'administration du cours. Pour chaque type d'activité, il est possible de définir les critères de réalisation. Il est également possible de combiner cet outil avec un plugin

9 Page resource, juin 2023, [https://docs.moodle.org/31/en/Page\\_resource](https://docs.moodle.org/31/en/Page_resource)

10 Catalogue, juin 2023, <https://docs.moodle.org/31/en/blocks/catalogue>

11 Reports, juin 2023, <https://docs.moodle.org/dev/Reports>

12 Carnet de notes, juin 2023, [https://docs.moodle.org/4x/fr/Carnet\\_de\\_notes](https://docs.moodle.org/4x/fr/Carnet_de_notes)

13 Achèvement des activités, juin 2023, [https://docs.moodle.org/3x/fr/Achèvement\\_des\\_activités](https://docs.moodle.org/3x/fr/Achèvement_des_activités)

supplémentaire appelé barre de progression<sup>14</sup>. Cette barre de progression est un outil visuel qui permet aux étudiants de suivre leur avancement dans la liste des tâches à accomplir.

## **2.3 Inconvénients de Moodle**

Comme pour H5P, lorsque l'on crée des tests directement dans Moodle (par exemple un QCM), on doit tout faire manuellement. C'est très chronophage pour les professeurs.

A noter que le projet ACE dont on a parlé en introduction vise à corriger ce problème ou du moins à rendre automatique la création de quiz au format GIFT, importable dans moodle

Un autre problème se présente : seuls les étudiants qui ont les droits d'accès à Moodle peuvent accéder aux tests sans restrictions.

De plus, Moodle ne nous offre pas la possibilité d'enregistrer le temps de réponse et le nombre de tentatives pour chaque question. On est donc incapables de déterminer sur quelles questions les étudiants ont hésité, quelles questions sont difficiles pour eux et lesquelles sont faciles.

Autre inconvénient, Moodle ne propose pas encore de paramètres pour rendre le site web accessible. Par exemple, régler la taille et la couleur du texte, la couleur de l'arrière-plan, l'espacement entre les lignes, etc. L'absence de ces paramètres empêche les personnes ayant des problèmes de vision de faire confortablement les exercices.

Enfin, bien que Moodle propose des blocs pour écrire des feedbacks, ces commentaires sont généralement rédigés par les enseignants et envoyés aux étudiants. Par exemple, lorsqu'un étudiant effectue un test, les enseignants doivent retrouver les résultats pour ajouter des commentaires. Cela prend beaucoup de temps, il faudrait donc trouver un moyen de générer ces remarques de manière automatique.

---

14 Michael de Raadt, Progress Bar, juin 2023, [https://moodle.org/plugins/block\\_progress](https://moodle.org/plugins/block_progress)

### 3. WIMS

#### 3.1 Présentation générale WIMS

WIMS (Web Interactive Multipurpose Server) est une plateforme e-learning collaborative et open source, disponible sous licence GNU (licence publique générale)<sup>15</sup>. WIMS a été initialement créé par Xiao Gang (1951-2014)<sup>16</sup>. La première version du logiciel a été publiée en 1997.

WIMS offre un large panel d'exercices interactifs en ligne dans de nombreux domaines tels que les mathématiques, la chimie, la physique, la biologie, le français, l'anglais, et bien d'autres encore. WIMS est principalement utilisé en France<sup>17</sup>, et plus particulièrement dans le domaine des mathématiques, que ce soit au lycée ou durant les premières années de l'enseignement supérieur.

WIMS offre aussi une multitude de ressources pédagogiques pour les étudiants, visant à les accompagner dans leur apprentissage. Parmi ces ressources, on retrouve des exercices à correction automatique qui permettent aux étudiants de travailler à leur propre rythme.

En outre, WIMS propose des outils graphiques et de calcul puissants et conviviaux, offrant ainsi aux étudiants des fonctionnalités avancées pour leurs travaux et leurs analyses.

Pour soutenir l'encadrement des étudiants, WIMS propose également des classes virtuelles. Ces espaces de travail en ligne permettent aux enseignants de superviser les activités des élèves, d'évaluer les exercices grâce à la correction automatique et de consulter les notes obtenues.

En résumé, WIMS est une plateforme complète, regroupant des ressources pédagogiques variées, des outils interactifs et des fonctionnalités d'évaluation, pour faciliter l'apprentissage des étudiants.

#### 3.2 Fonctions des exercices de WIMS

L'enseignant peut créer ses propres exercices interactifs et les insérer ensuite dans des feuilles de travail. Ces exercices bénéficient automatiquement des fonctions standard de WIMS :

---

<sup>15</sup> <https://fr.wikipedia.org/wiki/WIMS>

<sup>16</sup> Xiao Gang, juin 2023, <https://sites.google.com/view/gangwims/xiao-et-wims>

<sup>17</sup> Carte interactive montrant où WIMS est utilisé, juin 2023,  
<http://downloadcenter.wimsedu.info/download/map/map2.html>

- Choisir le nombre d'exercices sur lesquels on veut travailler
- Sélection aléatoire des exercices
- Enregistrement des notes
- Correction automatique, ce qui facilite le processus d'évaluation
- Feedbacks personnalisés basés sur les réponses des apprenants, incluant des aides textuelles.

### 3.3 Exercices de syntaxe sur WIMS

Sur WIMS, afin de trouver des exercices de syntaxe, on utilise la barre de recherche. Il suffit de saisir « Syntaxe » pour obtenir les résultats correspondants. Par exemple, on peut trouver des exercices portant sur les fonctions liées au verbe.

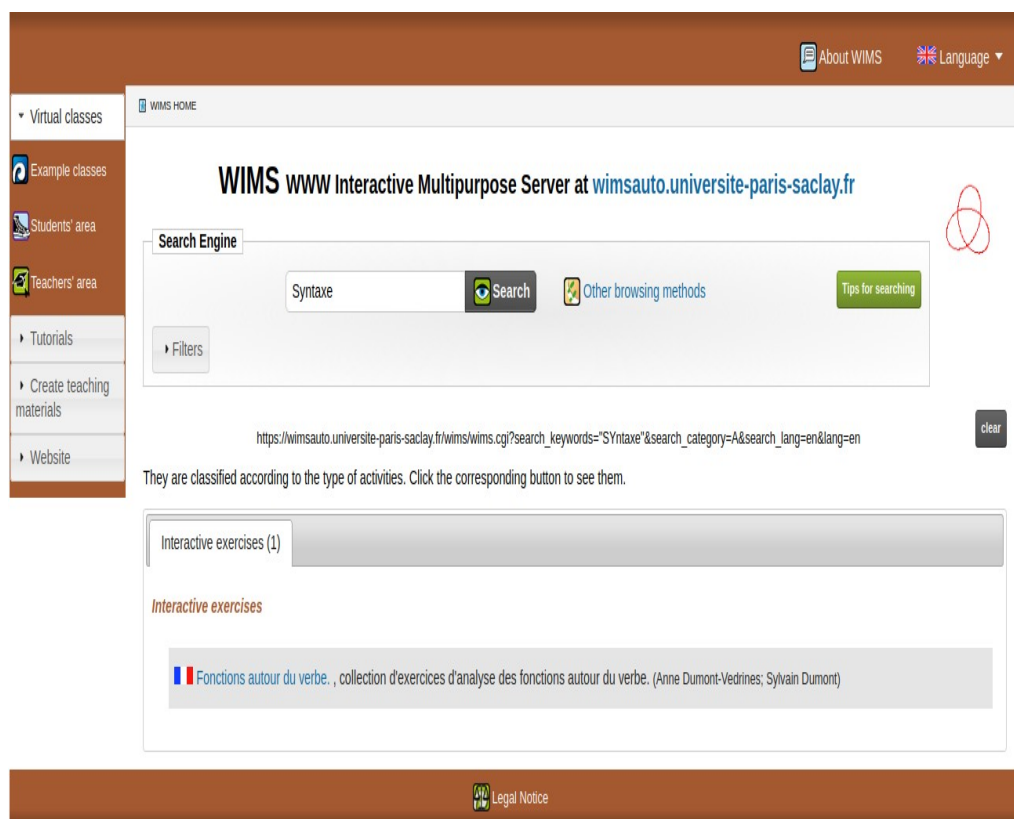


Figure 8: La page d'accueil de WIMS, où on peut taper ce qu'on veut chercher

Lorsque l'on clique sur « Fonctions autour du verbe », une nouvelle page s'ouvre, offrant diverses options pour choisir des exercices, paramétrer la série d'exercices, paramétrer l'analyse des réponses.

Tout d'abord, on a l'option de choisir les exercices qui constituent une série d'exercices. Dans cet exemple, le module propose une série de 7 exercices parmi laquelle on peut en sélectionner un ou plusieurs exercices.

Figure 9: Paramétrage : Choix des exercices

Ensuite, nous avons les paramètres des séries d'exercices. Voici la liste de ces options :

- Nombre d'exercices dans la série<sup>18</sup> : Déterminez le nombre d'exercices auxquels l'élève doit répondre. Si l'on fixe une série comportant plusieurs exercices, cela peut avoir des effets contradictoires. D'un côté, cela peut décourager l'élève en ajoutant une pression supplémentaire liée aux résultats obtenus. D'un autre côté, cela peut également restreindre sa possibilité de recommencer un exercice qui ne lui plaît pas, car il doit d'abord terminer la série imposée.
- Fixer l'ordre des exercices de la série<sup>19</sup> : Choisir un ordre aléatoire ou fixe. Cette fonctionnalité nous permet de proposer plusieurs variantes d'exercices de manière progressive.
- Maximum d'essais<sup>20</sup> : Définir le nombre maximum d'essais autorisés avec la même version de la série d'exercices. En permettant un certain nombre d'essais, les

18 Nombre d'exercices dans la série, Paramétrer l'exercice, juin 2023, <https://wims.univ-cotedazur.fr/wims/wims.cgi?session=DN00FDA16F.2&+lang=en&+module=help%2Fteacher%2Fdocbeginner.fr&+cmd=reply&+job=read&+foldin=30..4#primi.4>

19 Fixez l'ordre des exercices de la série, Paramétrer l'exercice, juin 2023, <https://wims.univ-cotedazur.fr/wims/wims.cgi?session=DN00FDA16F.2&+lang=en&+module=help%2Fteacher%2Fdocbeginner.fr&+cmd=reply&+job=read&+foldin=30..4#primi.4>

20 Maximum d'essais avec la même version de la série d'exercices, Paramétrer l'exercice, juin 2023, <https://wims.univ-cotedazur.fr/wims/wims.cgi?session=DN00FDA16F.2&+lang=en&+module=help%2Fteacher%2Fdocbeginner.fr&+cmd=reply&+job=read&+foldin=30..4#primi.4>

apprenants ont l'occasion de s'améliorer au fil du temps. Ils peuvent analyser leurs erreurs, apprendre de leurs erreurs précédentes et apporter les corrections nécessaires. Cela favorise un processus d'apprentissage progressif et itératif.

Figure 10: Paramétrage des séries d'exercices

Comparé à Moodle et H5P, WIMS offre aux étudiants la possibilité de choisir les exercices sur lesquels ils souhaitent travailler et de configurer différents paramètres. Avec WIMS, l'utilisateur peut sélectionner l'ordre des exercices, qu'il soit aléatoire ou fixe, ainsi que définir le nombre d'essais. Cela donne l'impression d'un jeu où le joueur peut personnaliser ses choix de jeu, ce qui encourage davantage à participer et rend l'utilisateur plus autonome.

### 3.4 Expérience personnelle de réalisation d'exercices sur WIMS.

Tout d'abord il faut savoir qu'il est possible de créer sur WIMS des exercices à partir d'une banque de données existantes, ou en mode « libre ». Pour essayer la plateforme nous avons décidé pour gagner du temps de tester la création d'exercice en utilisant des données existantes.

Avant de faire l'exercice, nous avons défini nos paramètres d'exercice. Tout d'abord, nous avons sélectionné un exercice ressemblant à ce que nous souhaitons mettre en place dans notre projet, et qui se focalise donc sur la différence entre le sujet, le COD et l'attribut du sujet.

Ensuite, nous avons opté pour une série d'exercices aléatoires pour faire varier les types d'exercices. Chaque exercice est paramétrable avec une limite de temps de 300 secondes maximum, et nous pouvons paramétrer le nombre de tentatives à 4 au maximum. Enfin, nous avons réglé le niveau de difficulté à 2, ce qui correspond à une difficulté modérée.

**Set-up**

Choose one or more exercises from the list and specify the settings (simplified or expert menu). Then click **Start**. The exercises will be randomly selected from your selection (or otherwise from among all the available exercises if you didn't select any).

**General parameters**

**Choose the exercises:**

Distinguer COD; COI; COS  
Distinguer sujet et COD.  
Distinguer sujet; COD; attribut du sujet  
Identifier les C. Circonstanciels  
Identifier les fonctions autour du verbe  
Les fonctions autour du verbe: le mix !  
Toutes les fonctions hors CC

**Select all exercises containing:**

**General setup**

**One series will have:**  exercises.

A series of exercises corresponds to the work that needs to be done before getting a grade. The order is by default the order you see in the list. It is only relevant if the number of selected exercises equals the number of exercises in the series.

☐ **Propose the exercises of the series in order.**  
You may later change this order after insertion in a worksheet.

**Time limit:**  seconds.  
You can put two numbers into the time limit, a smaller limit followed by a larger one, separated by a comma, without space. In this case, the first limit (counted in seconds) starts score reduction. The second number, by default equal to the first one, gives the delay until the score reaches 0.

**Maximum tries with the same version of an exercise series** ☐ 0 ☐ 1 ☐ 2 ☐ 3 ☒ 4 ☐ 5 ☐ 6  
Select a number n larger than 1 to avoid a change of the random data of the series at each new try: the data will change only after a correct answer or after n tries.

**Expert menu**

**Set-up for the analysis of answers**

**Level of severity:**

Click on **Expert menu** for further details.

Figure 11: Exercice fonctions autour du verbe sur WIMS



## Distinguer sujet, COD, attribut du sujet

Attention! This exercise is limited in time. 04:50

Les mots en gras sont-ils sujet, COD, ou attribut du sujet ? Cliquez sur la case correspondante.

Les gens honnêtes n'utilisent pas **de méthodes pernicieuses**. =>

?

sujet

COD

attribut du sujet

?

Submit answer(s)

Abandon

Figure 12: Première question: Indiquer le sujet, le COD et l'attribut du sujet

Après avoir sélectionné les paramètres, une page avec les exercices s'affiche. Ils comportent une consigne, une question, des réponses et un temps limité.

WIMS propose plusieurs types d'exercices différents pour distinguer le sujet, le COD, le COI, le COS et l'attribut du sujet :

- Glisser-déposer.
- Remplir le vide.
- Cliquer sur la case

## Distinguer COD, COI, COS

Indiquez la fonction des mots surlignés: sont-ils COD (Qu'est-ce que + S + V ?)? COI (A qui/quoi/De qui/quoi est-ce que + S + V ?)? COS (=COI quand il y a un autre CO dans la phrase)?

La principale a promis une belle récompense aux élèves habillés en mauve.

Figure 13: Remplir le vide: indiquer la fonction des mots surlignés.

La variété des types d'exercices permet aux étudiants de ne pas s'ennuyer lorsqu'ils travaillent.

### 3.5 Commentaires sur les consignes des exercices de syntaxe

Dans la consigne « *Indiquer la fonction des mots surlignés, soit COD, soit COI, soit COS (Complément d'Objet Second)* », on observe les termes COD et COI qui sont couramment utilisés dans les cours de français, y compris au lycée et à l'université. En revanche, le terme COS n'est généralement pas enseigné à l'université. Pour les étudiants il peut être difficile de choisir la bonne réponse lorsqu'ils rencontrent le terme « COS ». C'est pourquoi il est important de sélectionner des termes que les étudiants connaissent déjà ou de donner une explication claire pour éviter qu'ils rencontrent des problèmes lors de leurs exercices.

#### 3.5.1 Différents types de phrases

WIMS offre également différents types de phrases. En effet, certaines phrases ne suivent pas toujours l'ordre habituel du sujet suivi du verbe. Il existe également des exercices où les sujets sont placés après le verbe. Par exemple : « *Nombres sont aussi les possibilités de se racheter.* » Dans cette phrase, si un étudiant applique la règle selon laquelle « *un attribut du sujet est toujours placé après un verbe d'état* », il choisira la réponse « *attribut du sujet* ». Cependant, dans ce cas, le sujet est inversé, c'est-à-dire qu'il est placé après le verbe. Cet exercice peut poser des difficultés aux étudiants. Néanmoins, ces difficultés permettent aux étudiants de s'adapter à différents types d'exercices et de renforcer leurs compétences.

**Distinguer sujet, COD, attribut du sujet**

Les mots en gras sont-ils sujet, COD, ou attribut du sujet ? Cliquez sur la case correspondante.

Nombreuses sont aussi **les possibilités de se racheter.** => **sujet**

Figure 14: Un exemple dans la série: *Fontion autour de verbe*

### 3.5.2 Exercices théoriques

WIMS propose non seulement des exercices pratiques sur la construction des phrases, mais il offre également des exercices théoriques. Par exemple, la question suivante : « *Quelle question pose-t-on pour reconnaître un complément circonstanciel de temps ?* » Même si ce type d'exercice peut dans certains cas être utile aux étudiants, pour cet exemple précis il induit plutôt les étudiants en erreur, leur donnant de mauvaises habitudes, car ce type d'analyse (sémantique) n'est pas aussi fiable qu'une analyse syntaxique.

#### Identifier les fonctions autour du verbe

Cliquez sur la question à poser pour identifier chacune des fonctions autour du verbe.

Quelle question pose-t-on pour reconnaître un complément circonstanciel de temps ?  
 => ?

dans quel but (+ sujet + verbe) ?	par quel moyen (+ sujet + verbe à l'actif) ?	quand/combien de temps/depuis combien de temps (+ sujet + verbe) ?	par qui/par quoi (+ sujet + verbe au passif) ?	qui est-ce QUI/qu'est-ce QUI + verbe ?	à qui ?/à quoi ?/de qui ?/de quoi (+ sujet + verbe) ?
où (+ sujet + verbe) ?	qui est-ce QUE/qu'est-ce QUE + sujet + verbe attributif ?	de quelle manière (+ sujet + verbe) ?	qui est-ce QUE/qu'est-ce QUE (+ sujet + verbe) ?	à cause de quoi (+ sujet + verbe) ?	?

Figure 15: Un autre exemple dans la série: Fonction autour du verbe

### 3.5.3 Feedbacks

Les exercices proposés peuvent inclure des feedbacks (retour d'information) qui dépendent des réponses de l'apprenant. Cependant, ces feedbacks sont souvent limités à simplement dire si la réponse est une bonne réponse ou une mauvaise réponse, sans expliquer quelle est l'erreur. Les étudiants doivent donc trouver d'eux-mêmes les raisons de leur erreur. Il serait intéressant d'améliorer ces retours en expliquant clairement et brièvement les erreurs, afin d'aider les étudiants à comprendre leurs points faibles et à s'améliorer dans leur apprentissage de façon plus efficace.

## Distinguer COD, COI, COS

Indiquez la fonction des mots surlignés: sont-ils COD (Qu'est-ce que + S + V ?)? COI (A qui/quoi/De quoi/quoi est-ce que + S + V ?)? COS (=COI quand il y a un autre CO dans la phrase)?

On remettra ensuite des chaussures trouées aux jeunes sportifs qui seront arrivés en tête.

des COD [1] aux COI [2]

☐

Analyse de votre réponse

[1]	COD bonne réponse.
[2]	COI mauvaise réponse, la bonne réponse est cos.

Figure 16: Exemple avec le feedbacks

### 3.6 Conclusion concernant la partie de l'application WIMS.

En résumé, WIMS propose une grande variété d'exercices, de types d'exercices qui permettent aux étudiants de s'entraîner efficacement, même lorsqu'ils travaillent seuls à la maison. De plus, on peut utiliser WIMS sans créer de compte, ce qui signifie qu'on peut aller sur le site et faire les tests directement.

Cependant, il y a encore des aspects de WIMS qui nécessitent des améliorations. Par exemple, la création des exercices est actuellement réalisée manuellement, ce qui demande aux professeurs de consacrer du temps à apprendre et comprendre comment créer un exercice selon les exigences du site WIMS. Cette tâche peut prendre du temps. L'objectif de notre mémoire est de trouver un moyen de créer des exercices de manière automatique ou semi-automatique, afin de faciliter le processus pour les enseignants.

De plus, il est important d'améliorer la mise en page des exercices sur WIMS. En effet, la mise en page actuelle comprend une distinction visuelle claire entre les différentes sections : la partie consigne, la partie de la phrase, les questions et les boutons. Cependant, il convient de noter que le site actuel n'est pas accessible aux étudiants ayant des problèmes de vue. Par exemple, la combinaison de couleurs entre le fond sombre (vert) et la couleur d'écriture (noir)

ou encore la taille de police rend la lecture difficile. Il serait préférable d'adopter un fond clair avec un texte foncé afin de faciliter la lisibilité pour les personnes ayant des problèmes de vue, et permettre à l'utilisateur d'augmenter la taille de la police.

#### **4. Conclusion concernant l'état de l'art sur H5P, MOODLE et WIMS**

En résumé, H5P, Moodle et WIMS sont des plateformes gratuites. H5P fournit des outils pour la création de différents types d'exercices. Moodle et WIMS offre aux éducateurs et aux apprenants un environnement en ligne interactif pour l'apprentissage ainsi que des outils d'évaluation (les quiz, les sondages, etc), la gestion des ressources (des documents, des liens, des fichiers multimédias), la gestion de cours (gérer des cours en ligne), etc.

Cependant, chacune de ces trois plateformes présente des inconvénients. Les exercices doivent être créés manuellement, ce qui peut prendre énormément de temps. De plus, l'accessibilité est limitée, surtout pour les gens ayant des problèmes de vue. C'est pour cela que pour nous avons décidé de développer notre propre plateforme pour notre projet, et de ne pas réutiliser H5P, Moodle ou encore WIMS. Cependant, nous allons bien évidemment nous baser sur l'énorme travail déjà réalisé par ces différentes plateformes, et prendre des idées de chacune d'entre elle, auxquelles viendront s'ajouter les nôtres. Par exemple : questions aléatoires, feed-back détaillé, chronomètre...

### **III - DÉPENDANCE SYNTAXIQUE**

#### **1. Qu'est-ce que la dépendance syntaxique ?**

Ces dernières années, les représentations en dépendance sont devenues de plus en plus populaire dans l'analyse syntaxique.

L'analyse des dépendances consiste à identifier les relations de dépendance linguistique entre les mots « principaux » et les mots qui les modifient<sup>21</sup>. La dépendance syntaxique décrit les relations de subordination et de gouvernement entre les mots au sein d'une phrase. Contrairement à la grammaire traditionnelle basée sur les constituants (groupes de mots), la

---

21 « *Dependency parsing is the task of extracting a dependency parse of a sentence that represents its grammatical structure and defines the relationships between “head” words and words, which modify those heads.* » sur le site [NLP progress](https://nlp-progress.github.io/)

dépendance syntaxique se concentre sur la manière dont les mots se lient les uns aux autres dans une phrase, formant ainsi un graphe de dépendances.

Dans une phrase, un mot est généralement considéré comme la « tête » ou le « noyau », tandis que les autres mots sont considérés comme ses « dépendants ». La tête d'une phrase est souvent un verbe, mais elle peut également être un nom, un adjectif ou d'autres parties du discours. Les dépendants sont les mots ou les groupes de mots qui sont liés à la tête selon des relations grammaticales spécifiques. Ces relations de dépendance peuvent être de différents types, tels que le sujet, l'objet, le complément, le modifieur, etc. Ces relations de dépendance nous permettent de mieux comprendre la structure syntaxique des phrases et d'analyser leur signification plus précisément.

Le résultat d'un analyseur de dépendances est représenté par un graphe de dépendances, où les mots de la phrase d'entrée sont connectés par des relations de dépendance spécifiées<sup>22</sup>.

Voci un exemple sur la dépendance syntaxique : « *Clément trouve Nicolas bizarre.* »

Dans cette phrase, « trouve » est le verbe principal et agit comme la tête (root) de la phrase. Les dépendances syntaxiques sont les suivantes :

1. « Clément » est le sujet du verbe « trouve » et dépend du verbe « trouve ». La dépendance est de type sujet.
2. « Nicolas » est l'objet direct du verbe « trouve » et dépend également de ce verbe. La dépendance est de type objet direct.
3. « bizarre » est un adjectif attribut de l'objet direct « Nicolas ». Il dépend du sujet « Nicolas » et la dépendance est de type attribut du complément objet direct.

Afin de représenter ces dépendances sous forme d'arbre syntaxique, nous utilisons l'outil UDPipe pour créer une structure arborescente telle que celle-ci :

---

22 Lecture Notes: Part IV, *Dependency Parsing* <https://web.stanford.edu/class/cs224n/readings/cs224n-2019-notes04-dependencyparsing.pdf>, Christopher Manning, Richard Soche

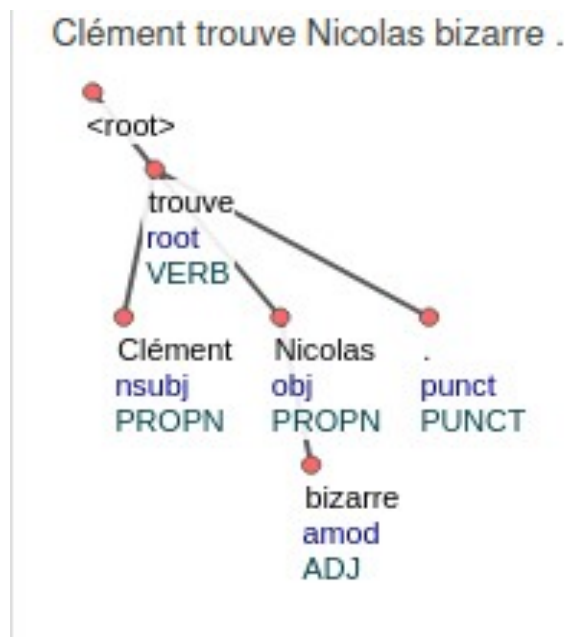


Figure 17: Capture d'écran de la dépendance syntaxique sur le site <https://lindat.mff.cuni.cz/services/udpipe/>

Cet arbre de dépendances représente comment chaque mot est lié à la tête (verbe) dans la phrase. Cette représentation permet de comprendre la structure syntaxique de la phrase de manière plus claire et précise.

### Pourquoi avons-nous besoin de la dépendance syntaxique ?

La dépendance syntaxique nous permet d'extraire des informations spécifiques d'une phrase donnée. Elle peut être utilisée pour identifier les relations entre les mots tels que déterminer le sujet, l'objet et d'autres éléments importants d'une phrase. Par exemple, dans la phrase : « *Ce livre est intéressant.* » *Ce livre* est le sujet de la phrase, tandis que *intéressant* est un attribut du sujet.

De plus, la dépendance syntaxique joue aussi un rôle dans la compréhension de la coréférence entre les noms et les pronoms compléments d'objet direct (COD) ou indirect (COI) au sein d'une phrase ou même entre des phrases. Reprenons les phrases suivantes : : « *Tu regardes **la télévision** ? Oui, je **la regarde** depuis une heure.* » Ici, nous pouvons comprendre que le pronom COD « la » dans la deuxième phrase fait référence à « *la télévision* » dont nous avons parlé dans la première phrase. De même, dans l'exemple avec COI : « *Je dois **téléphoner à mon mari**. Je vais **lui demander** de venir me chercher à la gare.* ». Nous comprenons que le

pronom COI «*lui*» dans la deuxième phrase fait référence au « *mon mari* » dont il est question dans la première phrase.

Ensuite, la dépendance syntaxique nous permet de résoudre les ambiguïtés et parvenir à une interprétation précise du sens global de la phrase. Prenons l'exemple suivant :

« *J'ai vu une fille avec des jumelles.* »

Dans cette phrase, l'ambiguïté réside dans l'interprétation du mot « avec », qui peut avoir deux significations différentes :

1. « *J'ai vu une fille qui avait des jumelles avec elle.* » Dans cette interprétation, la fille possède des jumelles. Ainsi, « *fille* » est le mot principal (ou « head ») de « *jumelles* », et « *jumelles* » est un modifieur de « *fille* ».
2. « *J'ai vu une fille en utilisant des jumelles.* » Dans cette interprétation, c'est la personne qui utilise les jumelles pour voir quelque chose. Donc, le mot « *vu* » est le mot principal de « *jumelles* », et « *jumelles* » est un modifieur de « *vu* ».

En comprenant les relations entre les mots, en particulier la dépendance entre « *fille* » et « *avec* », on peut déterminer quelle interprétation est la plus appropriée dans le contexte donné. La compréhension de cette relation permet de clarifier le sens de la phrase et d'éviter les malentendus.

La dépendance syntaxique est au cœur de notre projet. Dans le but de créer nos exercices de manière automatique, nous allons avoir besoin d'extraire des données comme les relations entre le noyau et ses dépendants. Ces données nous permettront de créer des exercices comme : trouver la partie de discours ou encore trouver la fonction grammaticale.

Nous allons maintenant aborder les outils et projets existants, et permettant justement d'extraire des données syntaxiques à partir d'un corpus.



## 2. Universal Dependencies (UD)

Universal Dependencies (UD) est un projet collaboratif qui a pour objectif de développer un framework (une structure logicielle) commun permettant de décrire de manière cohérente la grammaire des langues humaines à travers l'annotation des parties du discours, des caractéristiques morphologiques et les relations de dépendances syntaxiques (Nivre et al., 2016). Le projet a débuté en 2014 et s'est développé rapidement, avec plus de plus de 300 contributeurs qui ont produit près de 200 treebanks pour plus de 100 langues<sup>23</sup>. UD nous permet de télécharger des corpus dans n'importe quelle langue disponible sur la plateforme. UD fournit une liste de 37 relations<sup>24</sup> de dépendance universelle qui sont utilisées comme un standard pour annoter des arbres de dépendance syntaxique dans différentes langues.

### 2.1 Format CoNLL-U

CoNLL-U<sup>25</sup> est le format standard pour l'annotation des dépendances treebanks dans divers cadres, tels que les dépendances universelles (UD) et les dépendances universelles syntaxiques de surface (SUD). Ce format se compose de dix champs séparés :

- ID : Il s'agit de l'index numérique attribué à chaque mot dans une phrase. Il commence à 1 pour chaque nouvelle phrase.
- FORM : Ce champ correspond à la forme textuelle du mot lui-même ou à un symbole de ponctuation.
- LEMMA : Le champ LEMMA contient le lemme ou la racine du mot.
- UPOS : UPOS Tag sont utilisés pour annoter les parties du discours dans toutes les langues du projet Universal Dependencies, y compris le français. On notera que les UPOS tag sont universels, par exemple la valeur ADP pourra selon le langage correspondre soit à une préposition, soit à une postposition. Dans le cas du français par exemple ça sera toujours une préposition.

Par exemple : ADJ (Adjectif), ADP (Adposition), PUNCT (Ponctuation) ADV (Adverbe), AUX (Verbe auxiliaire), SYM(Symbole), INTJ (Interjection) CONJ

---

23 « *Universal Dependencies (UD) is a framework for consistent annotation of grammar (parts of speech, morphological features, and syntactic dependencies) across different human languages. UD is an open community effort with over 300 contributors producing nearly 200 treebanks in over 100 languages.* », <https://universaldependencies.org/>

24 <https://universaldependencies.org/u/dep/all.html>

25 Des informations sur le format CoNLL-U Format <https://universaldependencies.org/format.html>

(Conjonction de coordination), NOUN (noun), DET (Déterminant), PRON (Pronom) NUM (Numéral), VERB (verbe), PART (Particule), SCONJ (Conjonction de subordination)<sup>26</sup>.

- XPOS : Le champ XPOS est utilisé pour étiqueter une catégorie grammaticale spécifique utilisée par certaines langues dans le projet « Universal Dependencies ». Si ces caractéristiques ne sont pas disponibles, le champ est souligné.
- FEATS : Ce champ contient une liste de caractéristiques morphologiques du mots (genre, nombre, temps, etc.). Si ces caractéristiques ne sont pas disponibles, le champ est souligné.
- HEAD : Ce champ indique l'indice du mot gouverneur (tête) auquel le mot actuel est rattaché. Il est représenté par une valeur ID si une tête existe, sinon il est représenté par zéro (0).
- DEPREL : Il s'agit de la relation de dépendance universelle entre le mot actuel et sa tête (ou la racine si HEAD = 0). Dans certains cas, une sous-catégorie spécifique à la langue peut être utilisée pour décrire la relation.
- DEPS : Ce champ représente un graphe de dépendance amélioré sous la forme d'une liste de paires head-deprel, fournissant des informations détaillées sur les dépendances.
- MISC : Ce champ est utilisé pour toute autre annotation ou information supplémentaire qui ne correspond pas aux autres champs spécifiés.

Dans notre mémoire, nous allons nous concentrer sur cinq champs essentiels : ID, FORM, UPOS-Tag, HEAD et DEPREL. Les corpus annotés utilisés en entrée par le projet ACE sont au format CoNLL-U. À partir de ce format, le développeur du projet ACE a extrait les champs FORM, UPOS-Tag et DEPREL pour avoir des données qui permettent de générer des exercices d'identification des parties du discours et de la fonction grammaticale. Les données finales (les exercices) sont au format *gift*, format utilisé pour insérer des données dans la banque de questions et créer des exercices automatiquement sur Moodle. Dans notre projet, nous allons aussi utiliser les fichiers gift générés dans le cadre du projet ACE pour la création d'un de nos exercices, comme nous le verrons en détail dans la partie technique.

---

<sup>26</sup><https://universaldependencies.org/u/pos/index.html>

## 2.2 Générer des graphiques

Universal Dependency propose aussi l’outil Arborator<sup>27</sup>. C’est un outil qui permet de convertir des données qui sont au format CoNLL en un graphique (arbre syntaxique). L’utilisation de cet outil est très facile car il suffit d’insérer directement nos données au format CoNLL sur le site et l’arbre va être généré directement. C’est un gros avantage pour les utilisateurs ayant peu de connaissances informatiques.

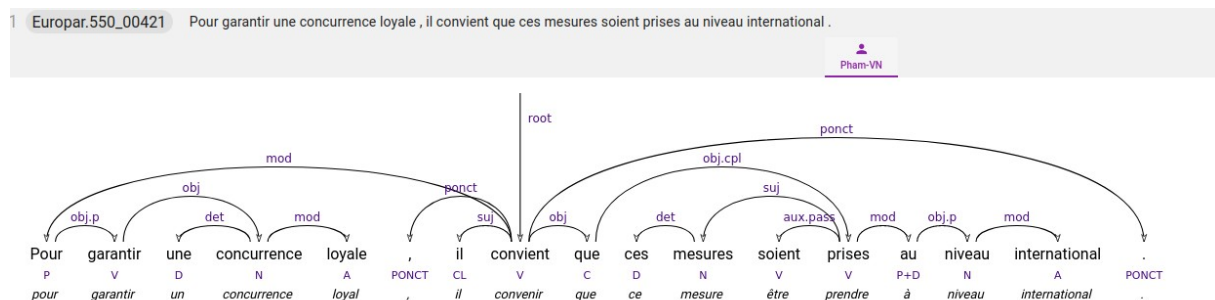


Figure 18: Une capture d'écran d'arbre syntaxique sur le site [Arborator.grew](http://Arborator.grew)

Comme le montre la figure ci-dessus, les données sont converties en un graphique avec un POSTag et des relations de dépendance universelles (DEPREL). Ce graphique permet de visualiser la relation entre les mots ainsi que la catégorie grammaticale de chaque mot. Ce sont des éléments dont on a besoin pour notre mémoire. De plus, ce site nous permet également de télécharger ces graphiques dans différents formats tels que SVG et CoNLL.

#### IV – ÉVALUATION DES CORPUS AVEC UDPIPE, SPACY, STANZA

## 1. Introduction

Lorsque l'on aborde le traitement automatique du langage, il est essentiel de mentionner les outils qui nous permettent d'annoter des corpus et de créer des arbres syntaxiques pour faciliter nos projets en TAL. Ainsi, dans cette partie, nous allons explorer trois outils majeurs que l'on a testé afin de déterminer lesquels utiliser pour notre besoin : Spacy, UDPipe et Stanza.

Tout d’abord, en tant qu’étudiant dans le domaine du traitement automatique des langues, il est certain que l’on a déjà eu une introduction à ces outils. Afin de mieux les connaître et d’acquérir la compétence nécessaire pour les utiliser efficacement dans des tâches de

27 <https://universaldependencies.org/tools.html#arborator>

linguistique telles que l’annotation de corpus ou l’analyse syntaxique, il est logique de se plonger davantage dans leur étude à travers ce mémoire.

Ensuite, en explorant ces outils en profondeur, nous pourrons non seulement en saisir les fonctions, mais aussi comprendre leur potentiel pour nos futurs projets de traitement des langues. En nous familiarisant avec les techniques et les capacités offertes par SpaCy, UDPipe et Stanza, nous pourrons aborder les défis linguistiques avec plus de confiance et d’efficacité.

Dans un second temps, nous procéderons à une évaluation approfondie de ces outils en ce qui concerne l’annotation de corpus. Cette évaluation nous permettra d’extraire et d’évaluer les données des champs UPOS (catégorie grammaticale), HEAD et DEPREL (relation universelle). Nous utiliserons deux corpus pour ces évaluations, un corpus qui se concentre sur l’attribut du COD, et un autre sur les compléments COD/COI. Ce sont des points de grammaire qui posent généralement problème aux étudiants lorsqu’ils effectuent des exercices.

L’évaluation de l’annotation des corpus nous aidera à choisir les meilleurs outils pour notre projet de création automatique d’exercices. Grâce à ces évaluations, nous pourrons aussi choisir entre travailler sur le corpus des attributs du COD, ou sur le corpus des compléments COD/COI, afin de mieux comprendre la fonction COD et COI. Cette démarche vise à générer automatiquement les exercices de façon efficace.

Les conclusions de ces évaluations nous aideront également à déterminer quelles catégories grammaticales exploiter dans les corpus COD et COI.

## **2. Présentation générale UDPipe**

UDPipe (Universal Dependencies Pipeline) est un outil open-source largement utilisé pour l’analyse syntaxique de textes dans différentes langues. Il fait partie du projet Universal Dependencies, qui vise à créer des schémas de dépendance cohérents et comparables pour de nombreuses langues. UDPipe effectue plusieurs tâches linguistiques comme la tokenisation, l’analyse morphologique, l’analyse syntaxique, la lemmatisation et l’étiquetage des parties du discours. Les résultats de l’analyse effectuée par UDPipe sont au format CoNLL-U d’Universal Dependencies.

UDPipe propose des modèles pré-entraînés pour différentes langues, ce qui permet d'obtenir des analyses syntaxiques et morphologiques sans avoir besoin de former un modèle spécifique à chaque fois.

### 3. Présentation générale Spacy

SpaCy est une bibliothèque logicielle open-source développée pour le traitement automatique du langage naturel (NLP) en Python<sup>28</sup>. Spacy propose des fonctionnalités telles que le traitement du texte, l'analyse syntaxique, l'étiquetage de parties du discours, la reconnaissance d'entités nommées, la désambiguïsation lexicale, etc.

Spacy est l'une des bibliothèques NLP les plus populaires et puissantes, largement utilisée parce qu'il est rapide, précis et prend en charge de plus de 73 langues<sup>29</sup>. De plus, Spacy peut traiter de grandes quantités de texte en un temps relativement court.

Spacy propose plusieurs fonctionnalités pour diverses tâches en traitement automatique du langage (TAL), telles que la tokenisation (découpage des textes en mots), l'étiquetage des parties du discours, l'analyse syntaxique (reconnaissance des relations entre les mots), la lemmatisation (réduction des mots à leur forme de base) et l'analyse des entités nommées (identification de noms de personnes, lieux, etc.)

Spacy est un outil plutôt orienté pour les entreprises, qui le choisissent notamment pour sa rapidité par rapport aux autres outils.

SpaCy offre aussi la possibilité de visualiser les arbres syntaxiques à l'aide de sa fonction `displacy`. Le visualiseur de dépendances de Spacy nous donne de différentes options de style (font texte, color de texte, la couleur de font...) pour la visualisation de l'arbre syntaxique<sup>30</sup>.

### 4. Présentation générale Stanza

Stanza est une librairie Python développée par le Stanford NLP Group et utilisée dans le traitement automatique des langues. Stanza est actuellement une solution en accès libre et gratuit. Stanza est capable de réaliser diverses analyses et annotations d'un texte telles que la tokenisation, la lemmatisation, l'annotation des parties du discours, des relations de

---

28 « *spaCy is a **free, open-source library** for advanced **Natural Language Processing (NLP)** in Python.* »  
What's Spacy <https://spacy.io/usage/spacy-101#whats-spacy>

29 « *Support for 73+ languages* » <https://spacy.io/>

30 <https://spacy.io/usage/visualizers>

dépendances ou encore la reconnaissance d'entités nommées. Stanza permet de travailler avec beaucoup de langues différentes et elle offre d'excellentes performances en qualité d'annotation et en vitesse de traitement.

Il propose un analyseur syntaxique, le Stanford Parser, qui utilise des modèles statistiques pour identifier les relations de dépendance entre les mots d'une phrase.

## 5. Évaluation du corpus contenant un attribut du COD

La première phrase que l'on va examiner est la suivante: «*Hélène juge ma cuisine insipide* ».

Voici les résultats obtenus :

# text = Hélène juge ma cuisine insipide.									
1	Hélène	Hélène	PROPN	_	_	2	nsubj	_	TokenRange=0:6
2	juge	juger	VERB	_	Mood=Ind Number=Sing Person=3 Tense=Pres VerbForm=Fin	0	root	_	TokenRange=7:11
3	ma	son	DET	_	Gender=Fem Number=Sing Number[psor]=Sing Person[psor]=1 Poss=Yes PronType=Prs	4	det	_	TokenRange=12:14
4	cuisine	cuisine	NOUN	_	Gender=Fem Number=Sing	2	obj	_	TokenRange=15:22
5	insipide	insipide	ADJ	_	Gender=Fem Number=Sing	4	amod	_	SpaceAfter=No TokenRange=23:31
6	.	.	PUNCT	_	_	2	punct	_	SpacesAfter='\r\n TokenRange=31:32

Figure 19: Capture d'écran de l'annotation d'un corpus avec l'outil UDPipe

ID: 0	TOKEN: Hélène	TAG: proper noun	Head: 0	DEPREL: root
ID: 1	TOKEN: juge	TAG: verb	Head: 0	DEPREL: None
ID: 2	TOKEN: ma	TAG: determiner	Head: 3	DEPREL: determiner
ID: 3	TOKEN: cuisine	TAG: noun	Head: 0	DEPREL: modifier of nominal
ID: 4	TOKEN: insipide	TAG: adjective	Head: 0	DEPREL: adjectival modifier
ID: 5	TOKEN: .	TAG: punctuation	Head: 0	DEPREL: punctuation

Figure 20: La capture d'écran du résultat obtenu en utilisant l'outil SpaCy

[Sentence 1]					
ID: 1	TOKEN: Hélène	LEMMA: Hélène	POS: PROPN	HEAD: 2	DEPREL: nsubj
ID: 2	TOKEN: juge	LEMMA: juger	POS: VERB	HEAD: 0	DEPREL: root
ID: 3	TOKEN: ma	LEMMA: son	POS: DET	HEAD: 4	DEPREL: det
ID: 4	TOKEN: cuisine	LEMMA: cuisine	POS: NOUN	HEAD: 2	DEPREL: obj
ID: 5	TOKEN: insipide	LEMMA: insipide	POS: ADJ	HEAD: 2	DEPREL: xcomp
ID: 6	TOKEN: .	LEMMA: .	POS: PUNCT	HEAD: 2	DEPREL: punct

Figure 21: La capture d'écran du résultat obtenu en utilisant l'outil Stanza

Afin de trouver la fonction de l'attribut du sujet dans la phrase « *Hélène juge ma cuisine insipide.* », nous pouvons utiliser la méthode de la substitution pronominale. Nous aurons donc : « *Elle la juge insipide.* ». Cette phrase peut s'analyser ainsi :

- Le pronom personnel complément « *la* » remplace le complément du COD (« *ma cuisine* »)
- L'adjectif « *insipide* » qualifie le COD « *cuisine* ». Cet adjectif se retrouve à droite du verbe principal « *juge* » lors de la substitution pronominale, donc il s'agit d'un attribut du COD.

Dans les résultats obtenus, UDPipe indique que le noyau de l'adjectif « *insipide* » est le nom « *cuisine* ». Cette analyse est incorrecte, car la tête de l'adjectif « *insipide* » doit être le verbe « *juge* ».

La relation de dépendance (DEPREL) entre l'adjectif « *insipide* » et le nom « *cuisine* » est donc un modifieur adjectival (amod) (voir la figure 20). En français, un modifieur adjectival peut être égal à une fonction épithète ou une fonction d'attribut du sujet ou du COD. Selon l'annotation de UD, cette analyse n'est pas fausse, car l'adjectif « *insipide* » qualifie le nom « *cuisine* » et la tête du mot « *insipide* » est le nom « *cuisine* ». Donc, le mot « *insipide* » est logiquement l'adjectif épithète (amod) du nom « *cuisine* ». Par contre, cela n'est pas le résultat que l'on cherche. La réponse que l'on souhaiterait est que l'adjectif « *insipide* » soit l'attribut du COD « *cuisine* ».

Par contre, Spacy montre que le noyau de l'adjectif « *insipide* » est le nom propre « *Hélène* ». En outre, le noyau du nom « *cuisine* » est aussi le nom « *Hélène* ». Ces analyses sont fausses. En effet, la tête de l'adjectif « *insipide* » et le nom « *cuisine* » doivent être le verbe « *juge* ».

De plus, selon des analyses de Spacy, la relation de dépendance (DEPREL) entre l'adjectif « *insipide* » et le nom « *Hélène* » est amod (soit l'adjectif épithète, soit l'adjectif l'attribut du sujet ou du COD). Ensuite, la relation de dépendance (DEPREL) entre le nom « *cuisine* » et le nom « *Hélène* » est un modifieur nominal (nmod = complément du nom). Toutes ces analyses sont fausses.



Dans le résultat fourni par Stanza, la tête de l'adjectif « *insipide* » est le verbe « *juge* », ce résultat est correct. Cependant, sa relation de dépendance universelle (DEPREL). D'après UD, « *xcomp* »<sup>31</sup> est un complément prédicatif d'un verbe ou d'un adjectif sans sujet propre. Ce-ci n'est pas correcte.

La deuxième phrase que l'on va examiner est la suivante : « *Max a trouvé ce vieux musicien hongrois génial* ».

Comme pour la phrase précédente, nous allons essayer de faire la pronominalisation pour identifier la fonction de l'adjectif « *génial* ». La phrase serait : « *Il l'a trouvé génial.* ». Dans cette phrase, on a :

- Le groupe nominal « *ce musicien* » qui est remplacé par le pronom personnel complément « *l* », il occupe la fonction de complément d'objet direct du verbe.
- L'adjectif « *génial* » qui qualifie le COD « *ce musicien* ». De plus, cet adjectif se retrouve à droite du verbe principal « *trouver* » lors de la substitution pronominale, donc l'adjectif « *génial* » est l'attribut du COD « *ce musicien* ».

---

31 « An open clausal complement (*xcomp*) of a verb or an adjective is a predicative or clausal complement without its own subject. The reference of the subject is necessarily determined by an argument external to the *xcomp* (normally by the object of the next higher clause, if there is one, or else by the subject of the next higher clause. These complements are always non-finite, and they are complements (arguments of the higher verb or adjective) rather than adjuncts/modifiers, such as a purpose clause. The name *xcomp* is borrowed from *Lexical-Functional Grammar*. », Stanford typed dependencies manual, Marie-Catherine de Marneffe and Christopher D. Manning, September 2008.

# text = Max a trouvé ce vieux musicien hongrois génial.

1	Max	Max	PROPN	_	_		3	nsubj
2	a	avoir	AUX	_	Mood=Ind Number=Sing Person=3 Tense=Pres VerbForm=Fin		3	aux:tense
3	trouvé	trouver	VERB	_	Gender=Masc Number=Sing Tense=Past VerbForm=Part		0	root
4	ce	ce	DET	_	Gender=Masc Number=Sing PronType=Dem		6	det
5	vieux	vieux	ADJ	_	Gender=Masc		6	amod
6	musicien	musicien	NOUN	_	Gender=Masc Number=Sing		3	obj
7	hongrois	hongrois	ADJ	_	Gender=Masc		6	amod
8	génial	génial	ADJ	_	Gender=Masc Number=Sing		6	amod
9	.	.	PUNCT	_	_		3	punct

Figure 22: Capture d'écran de l'annotation d'un corpus avec l'outil UDPipe

ID: 0	TOKEN: Max	TAG: proper noun	Head: 2	DEPREL: nominal subject
ID: 1	TOKEN: a	TAG: auxiliary	Head: 2	DEPREL: None
ID: 2	TOKEN: trouvé	TAG: verb	Head: 2	DEPREL: root
ID: 3	TOKEN: ce	TAG: determiner	Head: 5	DEPREL: determiner
ID: 4	TOKEN: vieux	TAG: adjective	Head: 5	DEPREL: adjectival modifier
ID: 5	TOKEN: musicien	TAG: noun	Head: 2	DEPREL: object
ID: 6	TOKEN: hongrois	TAG: adjective	Head: 5	DEPREL: adjectival modifier
ID: 7	TOKEN: génial	TAG: adjective	Head: 5	DEPREL: adjectival modifier
ID: 8	TOKEN: .	TAG: punctuation	Head: 2	DEPREL: punctuation

Figure 23: Capture d'écran de l'annotation d'un corpus avec l'outil Spacy

↳ [Sentence 1]

ID: 1	TOKEN: Max	LEMMA: Max	POS: PROPN	HEAD: 3	DEPREL: nsubj
ID: 2	TOKEN: a	LEMMA: avoir	POS: AUX	HEAD: 3	DEPREL: aux:tense
ID: 3	TOKEN: trouvé	LEMMA: trouver	POS: VERB	HEAD: 0	DEPREL: root
ID: 4	TOKEN: ce	LEMMA: ce	POS: DET	HEAD: 6	DEPREL: det
ID: 5	TOKEN: vieux	LEMMA: vieux	POS: ADJ	HEAD: 6	DEPREL: amod
ID: 6	TOKEN: musicien	LEMMA: musicien	POS: NOUN	HEAD: 3	DEPREL: obj
ID: 7	TOKEN: hongrois	LEMMA: hongrois	POS: ADJ	HEAD: 6	DEPREL: amod
ID: 8	TOKEN: génial	LEMMA: génial	POS: ADJ	HEAD: 6	DEPREL: amod
ID: 9	TOKEN: .	LEMMA: .	POS: PUNCT	HEAD: 3	DEPREL: punct

Figure 24: Capture d'écran de l'annotation d'un corpus avec l'outil Stanza.

Les trois résultats montrent que la catégorie grammaticale UPOS du mot « *génial* » est un adjectif. Cette analyse est correcte.

De plus, les trois résultats indiquent que la relation de dépendance entre l'adjectif « *génial* » et le nom « *musicien* » est un modifieur adjectival (amod = la fonction épithète ou la fonction d'attribut du sujet ou du COD). En outre, la tête du mot « *génial* » est le nom « *musicien* ». Cela signifie que l'adjectif « *génial* » est la fonction épithète du nom « *musicien* », car l'adjectif épithète ne peut pas dépendre d'un verbe. Si on est dans ce cas, cette analyse n'est pas correcte, car le verbe « *trouver* » est le noyau de l'adjectif « *génial* » et non le nom « *musicien* », donc sa relation est un attribut du COD, et non l'adjectif épithète.

La troisième phrase est : « *Max a appelé son chien Milo.* ». Après avoir utilisé la méthode de la substitution pronominale, la phrase serait : « *Il l'a appelé Milo.* ». En faisant la même analyse que pour les deux phrases précédentes, on constate que le nom propre « *Milo* » occupe la fonction d'attribut du complément direct.

# text = Max a appelé son chien Milo.									
1	Max	Max	PROPN	_	_		3	nsubj	_ TokenRange=0:3
2	a	avoir	AUX	_	Mood=Ind Number=Sing Person=3 Tense=Pres VerbForm=Fin		3	aux:tense	_ TokenRange=4:5
3	appelé	appeler	VERB	_	Gender=Masc Number=Sing Tense=Past VerbForm=Part		0	root	_ TokenRange=6:12
4	son	son	DET	_	Number=Sing Number[psor]=Sing Person[psor]=3 Poss=Yes PronType=Prs		5	det	_ TokenRange=13:16
5	chien	chien	NOUN	_	Gender=Masc Number=Sing		3	obj	_ TokenRange=17:22
6	Milo	Milo	PROPN	_	_		5	appos	_ SpaceAfter=No  TokenRange=23:27
7	.	.	PUNCT	_	_		3	punct	_ SpaceAfter=No  TokenRange=27:28

Figure 25: Capture d'écran de l'annotation d'un corpus avec l'outil UDPipe

ID: 0	TOKEN: Max	TAG: proper noun	Head: 2	DEPREL: nominal subject
ID: 1	TOKEN: a	TAG: auxiliary	Head: 2	DEPREL: None
ID: 2	TOKEN: appelé	TAG: verb	Head: 2	DEPREL: root
ID: 3	TOKEN: son	TAG: determiner	Head: 4	DEPREL: determiner
ID: 4	TOKEN: chien	TAG: noun	Head: 2	DEPREL: object
ID: 5	TOKEN: Milo	TAG: proper noun	Head: 4	DEPREL: adjectival modifier
ID: 6	TOKEN: .	TAG: punctuation	Head: 2	DEPREL: punctuation

Figure 26: Capture d'écran de l'annotation d'un corpus avec l'outil Spacy

[Sentence 1]					
ID: 1	TOKEN: Max	LEMMA: Max	POS: PROPN	HEAD: 3	DEPREL: nsubj
ID: 2	TOKEN: a	LEMMA: avoir	POS: AUX	HEAD: 3	DEPREL: aux:tense
ID: 3	TOKEN: appelé	LEMMA: appeler	POS: VERB	HEAD: 0	DEPREL: root
ID: 4	TOKEN: son	LEMMA: son	POS: DET	HEAD: 5	DEPREL: det
ID: 5	TOKEN: chien	LEMMA: chien	POS: NOUN	HEAD: 3	DEPREL: obj
ID: 6	TOKEN: Milo	LEMMA: Milo	POS: PROPN	HEAD: 5	DEPREL: appos
ID: 7	TOKEN: .	LEMMA: .	POS: PUNCT	HEAD: 3	DEPREL: punct

Figure 27: Capture d'écran de l'annotation d'un corpus avec l'outil Stanza

Les outils identifient que la catégorie grammaticale UPOS du mot «*Milo*» est un nom propre (PROPN).

De plus, les trois résultats montrent que le noyau du nom propre «*Milo*» est le nom «*chien*». Cependant, la relation de dépendance est différente pour les trois outils. Pour UDPipe et Stanza, «*Milo*» est un nom en apposition (appos). Tandis que pour Spacy c'est une relation du modifieur adjectival.

Ces résultats sont incorrects, car le nom propre «*Milo*» est l'attribut du COD «*son chien*». Sa tête devrait donc être le verbe «*appeler*».

## Conclusion du corpus contenant un attribut du COD

Pour le corpus contenant un attribut du COD, les trois outils ne donnent pas des résultats corrects pour les champs HEAD et DEPREL. Il ne semble donc pas judicieux de choisir ce corpus pour la génération des exercices.

Voici le tableau des résultats sur l'évaluation du corpus portant l'attribut du COD :

	UDPipe		Spacy		Stanza	
	HEAD	DEPREL	HEAD	DEPREL	HEAD	DEPREL
<i>Hélène juge ma cuisine insipide.</i>	<b>X</b>	<b>X</b>	<b>X</b>	<b>X</b>	✓	<b>X</b>
<i>Max a trouvé ce vieux musicien hongrois génial.</i>	<b>X</b>	<b>X</b>	<b>X</b>	<b>X</b>	<b>X</b>	<b>X</b>
<i>Max a appelé son chien Milo.</i>	<b>X</b>	<b>X</b>	<b>X</b>	<b>X</b>	<b>X</b>	<b>X</b>
Pourcentage de bons résultats	<b>0 %</b>		<b>0 %</b>		<b>16,6 %</b>	

Dans ce tableau, nous pouvons voir aussi que les résultats sont globalement mauvais. L'outil Stanza donne 16,6 % de bons résultats, les deux autres outils (UDPipe et Spacy) ne donnent aucun résultat correct.

## 6. Évaluation du corpus contenant un complément du COD ou un complément du COI

La première phrase est « *J'appelle Clément.* » où le nom propre « *Clément* » est le Complément d'Objet Direct (C.O.D.) du verbe « *appeler* ».

# text = J'appelle Clément.

1	J'	J'	PRON	_	_	2	nsubj	_	SpaceAfter=No TokenRange=0:2
2	appelle	appeler	VERB	_	Mood=Ind Number=Sing Person=3 Tense=Pres VerbForm=Fin	0	root	_	TokenRange=2:9
3	Clément	Clément	PROPN	_	_	2	obj	_	SpaceAfter=No TokenRange=10:17
4	.	.	PUNCT	_	_	2	punct	_	SpaceAfter=No TokenRange=17:18

Figure 28: Capture d'écran du corpus annoté «*J'appelle Clément*» avec UDPipe

ID: 1	TOKEN: J'	LEMMA: il	POS: PRON	HEAD: 2	DEPREL: nsubj
ID: 2	TOKEN: appelle	LEMMA: appeler	POS: VERB	HEAD: 0	DEPREL: root
ID: 3	TOKEN: Clément	LEMMA: Clément	POS: PROPN	HEAD: 2	DEPREL: obj
ID: 4	TOKEN: .	LEMMA: .	POS: PUNCT	HEAD: 2	DEPREL: punct

Figure 29: Capture d'écran du corpus annoté « *J'appelle Clément* » avec Stanza

ID: 0	TOKEN: J'	TAG: pronoun	Head: 1	DEPREL: nominal subject
ID: 1	TOKEN: appelle	TAG: verb	Head: 1	DEPREL: root
ID: 2	TOKEN: Clément	TAG: proper noun	Head: 1	DEPREL: object
ID: 3	TOKEN: .	TAG: punctuation	Head: 1	DEPREL: punctuation

Figure 30: Capture d'écran du corpus annoté « J'appelle Clément » avec Spacy.

En examinant les trois ensembles de résultats dans les colonnes UPOS, HEAD et DEPREL, on constate que les trois outils fournissent des résultats cohérents et exacts.

La deuxième phrase est « J'aime la musique. » où « musique » est le C.O.D. du verbe « aimer ».

# text = J'aime la musique.									
1	J'	il	PRON	_	Number=Sing Person=1 PronType=Prs	2	nsbj	_	SpaceAfter=No TokenRange=0:2
2	aime	aimer	VERB	_	Mood=Ind Number=Sing Person=1 Tense=Pres VerbForm=Fin	0	root	_	TokenRange=2:6
3	la	le	DET	_	Definite=Def Gender=Fem Number=Sing PronType=Art	4	det	_	TokenRange=7:9
4	musique	musique	NOUN	_	Gender=Fem Number=Sing	2	obj	_	SpaceAfter=No TokenRange=10:17
5	.	.	PUNCT	_		2	punct	_	SpaceAfter=No TokenRange=17:18

Figure 31: Capture d'écran du corpus annoté « J'aime la musique. » avec UDPipe



ID: 0	TOKEN: J'	TAG: pronoun	Head: 1	DEPREL: nominal subject
ID: 1	TOKEN: aime	TAG: verb	Head: 1	DEPREL: root
ID: 2	TOKEN: la	TAG: determiner	Head: 3	DEPREL: determiner
ID: 3	TOKEN: musique	TAG: noun	Head: 1	DEPREL: object
ID: 4	TOKEN: .	TAG: punctuation	Head: 1	DEPREL: punctuation

Figure 32: Capture d'écran du corpus annoté « J'aime la musique. » avec Spacy

[Sentence 1]					
ID: 1	TOKEN: J'	LEMMA: il	POS: PRON	HEAD: 2	DEPREL: nsubj
ID: 2	TOKEN: aime	LEMMA: aimer	POS: VERB	HEAD: 0	DEPREL: root
ID: 3	TOKEN: la	LEMMA: le	POS: DET	HEAD: 4	DEPREL: det
ID: 4	TOKEN: musique	LEMMA: musique	POS: NOUN	HEAD: 2	DEPREL: obj
ID: 5	TOKEN: .	LEMMA: .	POS: PUNCT	HEAD: 2	DEPREL: punct

Figure 33: Capture d'écran du corpus annoté « J'aime la musique. » avec Stanza

Les outils nous donnent des résultats corrects dans les trois colonnes UPOS, HEAD et DEPREL pour le nom « *musique* ».

La phrase suivante est : « *Je le mange.* » où le pronom « *le* » est le C.O.D. du verbe « *manger* ».

# text = Je le mange.							
1	Je	il	PRON	_	Number=Sing Person=1 PronType=Prs	3	nsubj
2	le	le	PRON	_	Gender=Masc Number=Sing Person=3 PronType=Prs	3	obj
3	mange	manger	VERB	_	Mood=Ind Number=Sing Person=1 Tense=Pres VerbForm=Fin	0	root
4	.	.	PUNCT	_	_	3	punct

Figure 34: Capture d'écran du corpus annoté « Je le mange. » avec UDPipe

ID: 0	TOKEN: Je	TAG: pronoun	Head: 2	DEPREL: nominal subject
ID: 1	TOKEN: le	TAG: determiner	Head: 2	DEPREL: object
ID: 2	TOKEN: mange	TAG: noun	Head: 2	DEPREL: root
ID: 3	TOKEN: .	TAG: punctuation	Head: 2	DEPREL: punctuation

Figure 35: Capture d'écran du corpus annoté « Je le mange. » avec Spacy.

[Sentence 1]					
ID: 1	TOKEN: Je	LEMMA: il	POS: PRON	HEAD: 3	DEPREL: nsubj
ID: 2	TOKEN: le	LEMMA: le	POS: PRON	HEAD: 3	DEPREL: obj
ID: 3	TOKEN: mange	LEMMA: manger	POS: VERB	HEAD: 0	DEPREL: root
ID: 4	TOKEN: .	LEMMA: .	POS: PUNCT	HEAD: 3	DEPREL: punct

Figure 36: Capture d'écran du corpus annoté « Je le mange. » avec Stanza

Dans cette phrase, UDPipe et Stanza indiquent que la catégorie grammaticale du mot « *le* » est un pronom. Ce résultat est correct. Par contre, Spacy ne donne pas un bon résultat, car pour Spacy, « *le* » est un déterminant.

Cependant, les trois outils nous fournissent des résultats corrects dans les deux colonnes HEAD et DEPREL. Le noyau du pronom personnel « *le* » est le verbe « *manger* ». Sa relation de dépendance est le complément d'objet direct.

La phrase suivante est: « *J'aime dessiner.* » On va vérifier si les trois outils nous donnent le résultat correct où «*dessiner*» est le C.O.D. du verbe «*aime*».

# text = J'aime dessiner.									
1	J'	il	PRON	_	Number=Sing Person=1 PronType=Prs	2	nsubj	_	
2	aime	aimer	VERB	_	Mood=Ind Number=Sing Person=1 Tense=Pres VerbForm=Fin	0	root	_	
3	dessiner	dessiner	VERB	_	VerbForm=Inf	2	xcomp	_	
4	.	.	PUNCT	_	-	2	punct	_	

Figure 37: Capture d'écran du corpus annoté «*J'aime dessiner.* » avec UDPipe

ID: 0	TOKEN: J'	TAG: pronoun	Head: 1	DEPREL: nsubj
ID: 1	TOKEN: aime	TAG: verb	Head: 1	DEPREL: ROOT
ID: 2	TOKEN: dessiner	TAG: verb	Head: 1	DEPREL: xcomp
ID: 3	TOKEN: .	TAG: punctuation	Head: 1	DEPREL: punct

Figure 38: Capture d'écran du corpus annoté «*J'aime dessiner.* » avec Spacy

```

-----
Phrase: J'aime dessiner.
Mot : J'      UPOS : PRON    HEAD : 2      DEPREL : nsubj
Mot : aime    UPOS : VERB    HEAD : 0      DEPREL : root
Mot : dessiner UPOS : VERB    HEAD : 2      DEPREL : xcomp
Mot : .       UPOS : PUNCT   HEAD : 2      DEPREL : punct

```

Figure 39: Capture d'écran du corpus annoté «*J'aime dessiner.* » avec Stanza

Dans cette phrase, les trois outils Stanza, UDPipe et Spacy indiquent que le verbe à l'infinitif « *dessiner* » dépend du verbe principal « *aimer* ». Sa relation de dépendance est « xcomp » (le complément prédiatif d'un verbe ou d'un adjectif). Le résultat n'est pas vraiment faux. Cependant, ce résultat n'est pas clair, car le complément du verbe peut être à la fois des C.O.D, des C.O.I. Donc, si on veut choisir le verbe à l'infinitif qui a la fonction COD ou COI, il est important de toujours effectuer une vérification minutieuse pour que le résultat soit clair et précis.

La phrase suivante est : « *Je me regarde dans le reflet de l'eau.* » où le pronom « *me* » est un COD du verbe « *regarder* ».

# text = Je me regarde dans le reflet de l'eau.

1	Je	il	PRON	_	Number=Sing Person=1 PronType=Prs	3	nsubj	_
2	me	se	PRON	_	Number=Sing Person=1 PronType=Prs Reflex=Yes	3	obj	_
3	regarde	regarder	VERB	_	Mood=Ind Number=Sing Person=1 Tense=Pres VerbForm=Fin	0	root	_
4	dans	dans	ADP	_	_	6	case	_
5	le	le	DET	_	Definite=Def Gender=Masc Number=Sing PronType=Art	6	det	_
6	reflet	reflet	NOUN	_	Gender=Masc Number=Sing	3	obl:mod	_
7	de	de	ADP	_	_	9	case	_
8	l'	le	DET	_	Definite=Def Number=Sing PronType=Art	9	det	_
9	eau	eau	NOUN	_	Gender=Fem Number=Sing	6	nmod	_
10	.	.	PUNCT	_	_	3	punct	_

Figure 40: Capture d'écran de l'annotation de la phrase "Je me regarde dans le reflet de l'eau." en utilisant la bibliothèque UDPipe

ID: 0	TOKEN: Je	TAG: pronoun	Head: 2	DEPREL: nsubj
ID: 1	TOKEN: me	TAG: pronoun	Head: 2	DEPREL: expl:comp
ID: 2	TOKEN: regarde	TAG: verb	Head: 2	DEPREL: ROOT
ID: 3	TOKEN: dans	TAG: adposition	Head: 5	DEPREL: case
ID: 4	TOKEN: le	TAG: determiner	Head: 5	DEPREL: det
ID: 5	TOKEN: reflet	TAG: noun	Head: 2	DEPREL: obl:arg
ID: 6	TOKEN: de	TAG: adposition	Head: 8	DEPREL: case
ID: 7	TOKEN: l'	TAG: determiner	Head: 8	DEPREL: det
ID: 8	TOKEN: eau	TAG: noun	Head: 5	DEPREL: nmod
ID: 9	TOKEN: .	TAG: punctuation	Head: 2	DEPREL: punct

Figure 41: Capture d'écran de l'annotation de la phrase "Je me regarde dans le reflet de l'eau." en utilisant la bibliothèque Spacy

Phrase : Je me regarde dans le reflet de l'eau.				
Mot : Je	Lemme : il	Pos : PRON	Head : 3	Relation : nsubj
Mot : me	Lemme : se	Pos : PRON	Head : 3	Relation : obj
Mot : regarde	Lemme : regarder	Pos : VERB	Head : 0	Relation : root
Mot : dans	Lemme : dans	Pos : ADP	Head : 6	Relation : case
Mot : le	Lemme : le	Pos : DET	Head : 6	Relation : det
Mot : reflet	Lemme : reflet	Pos : NOUN	Head : 3	Relation : obl:arg
Mot : de	Lemme : de	Pos : ADP	Head : 9	Relation : case
Mot : l'	Lemme : le	Pos : DET	Head : 9	Relation : det
Mot : eau	Lemme : eau	Pos : NOUN	Head : 6	Relation : nmod
Mot : .	Lemme : .	Pos : PUNCT	Head : 3	Relation : punct

Figure 42: Capture d'écran de l'annotation de la phrase "Je me regarde dans le reflet de l'eau." en utilisant la bibliothèque Stanza.

Stanza et UDPipe indiquent que le pronom « *me* » est un COD (obj) du verbe « *regarder* », ce résultat est correct. Par contre, pour Spacy la relation de dépendance entre le pronom « *me* » et le verbe « *regarder* » est étiquetée « *expl:comp* ». Cette relation est utilisée pour tous les pronoms réflexifs, ce qui n'est pas correct.

On va analyser la phrase suivante avec le pronom « *te* »: « *Je te ressemble.* » On va vérifier si les trois outils nous donnent le résultat correct où « *te* » est le C.O.I. du verbe « *ressemble* ».

# text = Je te ressemble.

1	Je	il	PRON	_	Number=Sing Person=1 PronType=Prs	3	nsubj	_
2	te	lui	PRON	_	Number=Sing Person=2 PronType=Prs	3	obj	_
3	ressemble	ressembler	VERB	_	Mood=Ind Number=Sing Person=1 Tense=Pres VerbForm=Fin	0	root	_
4	.	.	PUNCT	_	_	3	punct	_

Figure 43: Capture d'écran de l'annotation de la phrase "Je te ressemble." en utilisant la bibliothèque UDPipe

ID: 0	TOKEN: Je	TAG: pronoun	Head: 1	DEPREL: nsubj
ID: 1	TOKEN: te	TAG: verb	Head: 1	DEPREL: ROOT
ID: 2	TOKEN: ressemble	TAG: adjective	Head: 1	DEPREL: xcomp
ID: 3	TOKEN: .	TAG: punctuation	Head: 1	DEPREL: punct

Figure 44: Capture d'écran de l'annotation de la phrase "Je te ressemble." en utilisant la bibliothèque Spacy

Mot : Je	Lemme : il	Pos : PRON	Head : 3	Relation : nsubj
Mot : te	Lemme : lui	Pos : PRON	Head : 3	Relation : iobj
Mot : ressemble	Lemme : ressembler	Pos : VERB	Head : 0	Relation : root
Mot : .	Lemme : .	Pos : PUNCT	Head : 3	Relation : punct

Figure 45: Capture d'écran de l'annotation de la phrase "Je te ressemble." en utilisant la bibliothèque Stanza

Pour cette phrase, les trois outils fournissent des réponses différentes.

Selon les analyses de UDPipe, le mot « *te* » est un pronom. Sa tête est le verbe « *ressembler* ». Ces analyses sont correctes. Cependant, UDPipe identifie que le pronom « *te* » est un COD (obj) du verbe « *ressembler* ». Ce résultat n'est pas correct, car le pronom « *te* » est un complément du COI du verbe « *ressembler* ». (*Je te ressemble => Je lui ressemble*. Ici le pronom « *lui* » et « *te* » occupent la fonction du complément d'objet indirect).

Les analyses de Spacy ne sont pas correctes, car Spacy indique que :

- Le mot « *ressemble* » est un adjectif (voir la figure 45 : « *Tag : adjective* »). Cette analyse est totalement fausse. En fait, le mot « *ressemble* » est un verbe et il est le noyau.
- Le mot « *te* » est un verbe (« *Tag : verbe* ») et il est le noyau (« *root* »). Cette analyse est aussi fausse. En réalité, le mot « *te* » est un pronom et il n'est pas le noyau. Le noyau du pronom « *te* » est le verbe « *ressembler* ». Sa relation de dépendance est un complément d'objet indirect.

Les analyses de Stanza sont correctes, car Stanza indique que :

- Le noyau est le verbe « *ressembler* ».
- Le mot « *te* » est un pronom. Sa relation de dépendance avec sa tête « *ressembler* » est un COI (iobj).

On va ensuite examiner la phrase suivante avec le pronom « *en* »: « *Il s'en souvient* » On va vérifier si les trois outils nous donnent le bon résultat où « *en* » est le complément d'objet indirect (COI) du verbe « *souvient* ».

1	Il	il	PRON	_	Gender=Masc Number=Sing Person=3 PronType=Prs	4	nsubj	_
2	s'	s'	PRON	_	Person=3 PronType=Prs Reflex=Yes	4	expl:pv	_
3	en	en	PRON	_	Person=3 PronType=Prs	4	iobj	_
4	souvient	souvenir	VERB	_	Mood=Ind Number=Sing Person=3 Tense=Pres VerbForm=Fin	0	root	_
5	.	.	PUNCT	_	_	4	punct	_

Figure 46: Capture d'écran de l'annotation de la phrase "Il s'en souvient." en utilisant la bibliothèque UDPipe.



ID: 0	TOKEN: Il	TAG: pronoun	Head: 3	DEPREL: nsubj
ID: 1	TOKEN: s'	TAG: adverb	Head: 3	DEPREL: cop
ID: 2	TOKEN: en	TAG: adposition	Head: 3	DEPREL: case
ID: 3	TOKEN: souvient	TAG: verb	Head: 3	DEPREL: ROOT
ID: 4	TOKEN:	TAG: space	Head: 3	DEPREL: dep

Figure 47: Capture d'écran de l'annotation de la phrase "Il s'en souvient." en utilisant la bibliothèque Spacy.

Phrase : Il s'en souvient.				
Mot : Il	Lemme : il	Pos : PRON	Head : 4	Relation : nsubj
Mot : s'	Lemme : se	Pos : PRON	Head : 4	Relation : expl:pv
Mot : en	Lemme : en	Pos : PRON	Head : 4	Relation : iobj
Mot : souvient	Lemme : souvenir	Pos : VERB	Head : 0	Relation : root
Mot : .	Lemme : .	Pos : PUNCT	Head : 4	Relation : punct

Figure 48: Capture d'écran de l'annotation de la phrase "Il s'en souvient." en utilisant la bibliothèque Stanza.

Les trois outils Stanza, UDPipe et Spacy donnent des résultats divergents.

Stanza et UDPipe donnent le même résultat, indiquant que le pronom « en » est un Complément d'Objet Indirect (COI) du verbe « souvenir », ce qui est correct.

Spacy par contre génère un résultat où la relation DEPREL est étiquetée comme « case » (« case » est un élément marquant par exemple comme les prépositions.). Ce résultat est inexact, car le pronom « en » ici fonctionne comme un COI.

Lorsque l'on analyse la phrase « Je m'en moque. » en utilisant les trois outils UDPipe, Stanza et Spacy, on obtient le même résultat que pour la phrase « Il s'en souvient ».

On va examiner la phrase suivante avec le pronom « y »: « J'y pense. » On va vérifier si les trois outils nous donnent le bon résultat où le pronom « y » est le complément d'objet indirect (COI) du verbe « penser ».



# text = J'y pense.									
1	J'	il	PRON	_	Number=Sing Person=1 PronType=Prs	3	nsubj	_	SpacesBefore=\s SpaceAfter=No TokenRange=1:3
2	y	y	PRON	_	Person=3 PronType=Prs	3	iobj	_	TokenRange=3:4
3	pense	penser	VERB	_	Mood=Ind Number=Sing Person=1 Tense=Pres VerbForm=Fin	0	root	_	SpaceAfter=No TokenRange=5:10
4	.	.	PUNCT	_	_	3	punct	_	SpaceAfter=No TokenRange=10:11

Figure 49: Capture d'écran de l'annotation de la phrase "J'y pense." en utilisant la bibliothèque UDPipe.

ID: 0	TOKEN: J'	TAG: pronoun	Head: 2	DEPREL: nsubj
ID: 1	TOKEN: y	TAG: pronoun	Head: 2	DEPREL: iobj
ID: 2	TOKEN: pense	TAG: verb	Head: 2	DEPREL: ROOT
ID: 3	TOKEN: .	TAG: punctuation	Head: 2	DEPREL: punct

Figure 50: Capture d'écran de l'annotation de la phrase "J'y pense." en utilisant la bibliothèque Spacy.

Phrase : J'y pense.				
Mot : J'	Lemme : il	Pos : PRON	Head : 3	Relation : nsubj
Mot : y	Lemme : y	Pos : PRON	Head : 3	Relation : iobj
Mot : pense	Lemme : penser	Pos : VERB	Head : 0	Relation : root
Mot : .	Lemme : .	Pos : PUNCT	Head : 3	Relation : punct

Figure 51: Capture d'écran de l'annotation de la phrase "J'y pense." en utilisant la bibliothèque Stanza.

En observant les trois ensembles de résultats dans les colonnes DEPREL, on constate que les trois outils présentent des résultats cohérents et précis en ce qui concerne le pronom « y » qui est un COI du verbe « penser ».

## Conclusion pour l'évaluation de corpus contenant un complément du COD/COI

On constate que pour les corpus contenant des compléments COD/COI, les résultats sont bien meilleurs que pour le corpus qui contient des attributs du COD. De plus, on observe que c'est avec Stanza que l'on obtient les résultats les plus précis (92,8%). Pour UDPipe (85,7 %) les résultats sont légèrement moins bien mais restent tout de même très bons. Pour Spacy (64,2 %) par contre, les résultats sont très moyens (voir le tableau ci-dessous). Nous allons donc utiliser ce corpus pour générer nos exercices visant à reconnaître les fonctions du COD/COI

	UDPipe		Spacy		Stanza	
	HEAD	DEPREL	HEAD	DEPREL	HEAD	DEPREL
<i>J'appelle Clément.</i>	✓	✓	✓	✓	✓	✓
<i>J'aime la musique.</i>	✓	✓	✓	✓	✓	✓
<i>J'aime dessiner.</i>	✓	X	✓	X	✓	X
<i>Je me regarde dans le reflet de l'eau.</i>	✓	✓	✓	X	✓	✓
<i>Je te ressemble.</i>	✓	X	X	X	✓	✓
<i>Il s'en souvient.</i>	✓	✓	✓	X	✓	✓
<i>J'y pense.</i>	✓	✓	✓	✓	✓	✓
Pourcentage de bons résultats	85,7 %		64,2 %		92,8 %	

## 7. Conclusion de l'évaluation des corpus avec UDPIPE, Spacy, Stanza

À partir de l'analyse de ces corpus à l'aide des trois outils UDPipe, Spacy et Stanza, il apparaît que les performances d'UDPipe et de Stanza sont généralement plus fiables que celles de Spacy.

En utilisant UDPipe, nous avons la possibilité d'effectuer une annotation en ligne via le site <https://lindat.mff.cuni.cz/services/udpipe/>. Ce site nous permet de sélectionner la langue et le modèle d'annotation souhaité, générant rapidement des résultats au format CoNLL. De plus, le téléchargement au format CoNLL est simple et pratique. Nous avons également observé que le projet (ACE) de Monsieur Antonio Balvet a utilisé l'outil UDPipe pour annoter des corpus en vue de générer automatiquement des exercices.

Il est à noter qu'UDPipe met aussi une API à disposition des utilisateurs. Une API est une interface qui permet à plusieurs applications ou systèmes informatiques de communiquer entre eux. Dans notre cas cela nous permettrait directement depuis notre application d'appeler l'interface UDPipe, en spécifiant quel service on souhaite utiliser (par exemple le service *process*, qui permet de traiter un corpus). En retour nous récupérerons un fichier au format JSON (JavaScript Object Notation), qui est un format d'échange de données très utilisé, facile à lire et comprendre pour les humains, et facile à traiter/générer pour les machines.

UDPipe est un bon outil pour l'annotation. Malgré tout, pour la première version de notre projet présentée dans notre mémoire, nous avons décidé de ne pas l'utiliser pour les raisons suivantes.

Afin d'obtenir les données extraites au format CoNLL-U, nous devons aller sur le site de UDPipe pour exécuter l'analyse et récupérer son résultat. Cela prend du temps, surtout si on veut annoter un grand corpus. Nous pourrions utiliser l'API pour contourner ce problème, cependant pour la version initiale de notre projet nous n'avons mis en place qu'une application « locale », et pas de serveur web. Il ne nous est donc en l'état pas possible d'appeler l'API UDPipe depuis notre application.

Nous avons donc décidé d'utiliser Stanza, outil avec lequel les résultats sont tout aussi excellents. De plus, Stanza facilite l'annotation et l'extraction du corpus en fournissant des résultats au format JSON, qui seront utiles pour la création d'exercices. Cela nous permettra

de créer des exercices, notamment pour identifier les fonctions COD et COI (je détaillerai cette partie dans la section consacrée à la réalisation technique de mon mémoire).

Comme mentionné précédemment, en raison des résultats des évaluations des deux corpus, nous avons opté pour l'utilisation du corpus combiné du COD et du COI. De plus, nous avons fait ce choix car les étudiants ont souvent besoin de travailler sur la reconnaissance du COD et du COI plutôt que sur les attributs du COD.

## **V – RÉALISATION TECHNIQUE**

### **1. Présentation globale de projet**

Notre projet est composé de deux parties principales. La première partie est l'analyse des corpus et l'extraction des données dont nous avons besoin avec les outils Stanza et UDPipe, données que l'on transformera au format JSON. Ce fichier JSON nous permettra ensuite de générer un exercice de manière automatique.

La deuxième partie du projet est de la création d'une plate-forme en ligne (site web), qui permet d'afficher et d'effectuer les exercices sous la forme d'activité Quiz. Cette plate-forme est accessible à tout le monde, aucune inscription/authentification n'est nécessaire. De plus, elle permet aux étudiants de suivre leur progression tout au long de l'exercice, notamment à l'aide une barre de progression et d'un graphique. Enfin, cette plate-forme met à disposition de l'utilisateur toute une liste de paramètres d'accessibilité, que l'utilisateur peut modifier jusqu'à obtenir un affichage qui lui convient.

### **2. Choix du type d'exercice**

Pour nos exercices, nous avons choisi de faire des quiz (plus précisément des QCU):

Premièrement, l'utilisation d'un quiz améliore les performances d'apprentissage des étudiants. Les recherches de Jonathan Fernandez (2016) comparent les résultats de deux groupes : un groupe utilisant des quiz et un autre groupe qui n'en utilise pas. Les résultats montrent que les performances réelles du groupe utilisant des quiz étaient supérieures à celles du groupe qui n'en utilise pas. Les quiz sont en effet un excellent moyen d'encourager et de motiver les élèves à s'engager dans une tâche et à renforcer leur apprentissage.

Deuxièmement, nous avons vu dans notre analyse de l'état de l'art en ce qui concerne les exercices interactifs que la difficulté à générer des exercices de manière automatique dépend notamment du type d'exercice. Par exemple, un exercice de type *Find multiple Hotspots* (on clique sur plusieurs zones d'image pour trouver une bonne réponse.) est très difficile à automatiser en raison du besoin de détecter des réponses via les images. C'est pour cette raison que nous n'avons pas choisi ce type d'exercice.

Troisièmement, les QCU sont simples à comprendre et effectuer. Les étudiants ont l'habitude de travailler avec type d'exercice.

Nous aurions pu essayer de générer des exercices de type *glisser-déposer* ou encore *Mark the words*, cependant en raison du temps limité dont nous disposions, nous avons préféré nous orienter vers les QCM, plus simples à mettre en place et à automatiser. N'oublions pas que le but de notre mémoire était notamment de prouver l'on peut partir d'un corpus et arriver jusqu'à la création de l'exercice, le tout de manière automatique.

### **3. Choix de l'exercice**

#### **3.1 Choix de l'exercice sur la partie du discours**

Le premier exercice que nous avons choisi pour notre plateforme consiste à identifier les parties du discours (les catégories grammaticales). Nous avons choisi cet exercice pour plusieurs raisons.

Premièrement, la connaissance des catégories grammaticales est nécessaire pour l'acquisition de la syntaxe (Miller (1962))<sup>32</sup>. Elles aident à créer un grand nombre de phrases différentes et à distinguer celles qui sont correctes d'un point de vue grammatical de celles qui ne le sont pas.

Deuxièmement, lorsque l'on connaît la catégorie grammaticale, on peut analyser le fonctionnement de la langue et comment les phrases sont construites. Cela peut nous aider à améliorer les compétences linguistique. C'est un objectif central pour les étudiants en sciences du langage, car la maîtrise de la compétence linguistique est essentielle dans ce domaine.

Enfin, ce type d'exercice peut être utilisé lors des examens de début d'année qui servent aux professeurs à estimer le niveau des élèves, et aux élèves à avoir connaissances de leurs

---

<sup>32</sup> Jean-Adolphe Rondal, *Peut-on se dispenser des catégories grammaticales, des hiérarchies et des règles formelles dans l'acquisition de la syntaxe en langue naturelle ?* L'Année Psychologique, 2017/1 (Vol. 117), pages 1 à 40

potentielles lacunes. Ces exercices pourraient notamment être utilisés en auto-évaluation, ce qui serait un gain de temps non négligeable pour les professeurs.

### **3.2 Choix de l'exercice de la fonction grammaticale : COD et COI**

Le deuxième exercice que nous avons choisi d'intégrer porte sur les fonctions grammaticales. Cet exercice a été conçu pour les apprenants du français de niveau A2/B1. Nous avons décidé de créer cet exercice car, d'après notre expérience, l'identification des compléments d'objet direct (COD) et des compléments d'objet indirect (COI) pose des difficultés aux apprenants. Une recherche menée par A.Y.M.De-Souza et E.K.Bakah a révélé que 92 % des étudiants en FLE du département de français à l'université de Cape Coast rencontrent des problèmes liés à l'utilisation des pronoms personnels COD et COI (d'une part l'identification des COD/COI, d'autre part la place des pronoms personnels COD, COI).

Cette étude révèle que les difficultés des apprenants proviennent en grande partie du transfert négatif des structures de l'anglais vers le français, ou encore des différences de systèmes grammaticaux entre la langue maternelle de l'étudiant et le français. Certains enseignants considèrent également que la complexité des pronoms personnels COD/COI rend difficile pour les apprenants de saisir les notions de transitivité et d'intransitivité qui définissent les COD et les COI.

En tant qu'apprenant du français moi-même, je reconnais que la partie de grammaire relative aux COD et aux COI me pose toujours des difficultés. Par exemple, au lieu de dire « *Je lui parle.* » dans la phrase « *Je parle à mon père.* », on peut dire par erreur « *Je le parle.* ». Les problèmes liés à la pronominalisation des COD et des COI sont courants chez les étudiants étrangers. C'est pour cette raison que nous avons choisi l'exercice portant sur les COD et les COI, afin de permettre aux étudiants de s'entraîner. Si les étudiants parviennent à identifier la fonction des pronoms indiqués dans le quiz, cela signifie qu'ils ont compris les bases de la construction des COD et des COI.

## 4. Extraction des données pour alimenter notre application.

### 4.1 Choix technique

Notre choix s'est porté sur le langage Python pour plusieurs raisons. Il possède beaucoup de bibliothèques utiles pour le traitement automatique du langage naturel (TALN), telles que NLTK, spaCy, Stanford CoreNLP, et bien d'autres. Ces bibliothèques nous aident à faire des choses comme la tokenisation, l'analyse syntaxique, l'extraction des parties du discours et diverses autres opérations spécifiques au NLP. Dans notre travail, nous avons utilisé Stanza, qui est aussi une bibliothèque Python, pour étudier les relations de dépendance entre les mots. Cela nous a aidé à obtenir et formater les données nécessaires à la création des exercices portant sur les fonctions grammaticales. De plus, Python nous offre un grand nombre de ressources et de guides pratiques. Beaucoup de documentation est disponible, tant pour le langage en lui-même que pour les différentes bibliothèques mentionnées précédemment.

### 4.2 Préparation des données pour l'exercice Partie du discours (fichier QuizPartieDeDiscours)

Le projet ACE (Annotated Corpora for online Exercises) a pour objectif de créer des exercices d'auto-correction en utilisant un corpus annoté CONLL-U créé à l'aide de l'outil d'analyse syntaxique UDPIPE.

Dans notre projet, nous avons utilisé un des fichiers au format GIFT (*Sequoia-subset-quiz.gift*<sup>33</sup>) créé dans le cadre du projet ACE et que l'on peut trouver sur [github](https://github.com/abalvet/ACE/blob/main/v0.9/moodle-quizzes/Sequoia-subset-quiz.gift). Ce fichier a été utilisé pour créer un exercice portant spécifiquement sur la catégorie grammaticale des mots. Par la suite, en utilisant le langage Python, on a créé trois listes essentielles : une liste de phrases, une liste de questions et une liste de réponses. En combinant ces trois listes, nous avons créé une liste unique qui peut être convertie au format JSON. Ce fichier JSON a ensuite été intégré dans notre code JavaScript, ce qui nous a permis de générer l'exercice de manière automatique. Voici mon code pour illustrer le processus.

---

33 <https://github.com/abalvet/ACE/blob/main/v0.9/moodle-quizzes/Sequoia-subset-quiz.gift>

```

import re
import json

#Open and read file
fichier = open("Sequoia-subset-quiz.txt", 'rt')
lignes = fichier.readlines()

expressionOfPhrases = re.compile("[A-Za-z]")
expressionOfQuestions = re.compile("^[::]")

listeDeReponse = False

questions = list()
phrases = list()
reponse = list()

for line in lignes:

    #extraire des phrases
    if expressionOfPhrases.match(line):
        phrases.append(re.sub(r'\n', '', line))

    #extraire des questions
    if expressionOfQuestions.match(line):
        questions.append(re.sub(r'\n', '', line))

    #extraire des réponses
    if "}" in line:
        listeDeReponse = False

    if listeDeReponse == True:
        reponse.append(re.sub(r'\s', '', line))

    if "{" in line :
        listeDeReponse = True

#diviser une liste en mutiple listes
def divide_chunks(l, n):

    for i in range(0, len(l), n):
        yield l[i:i + n]

#chaque reponse et chaque phrase ont des ensembles de réponses
listOfquestion = list(divide_chunks(questions, 1))
listOfphrase = list(divide_chunks(phrases, 1))
listOfReponse = list(divide_chunks(reponse, 6))

listOfQuiz = []
trueCorrect = True
falseCorrect = False

```

Figure 52: Capture l'écran en illustrant de code Python (convertir file texte en JSON)



```

for i in range(0,len(listOfQuestion)):
    listOfReponses = []
    dictOfQuiz= {}

    # convertir une liste de réponse en string
    question = ' '.join(listOfQuestion[i])
    phrase = ' '.join(listOfphrase[i])

    # extraire des réponses pour chaque question et montrer une bonne réponse ou une mauvais réponse
    for j in range(0,6):
        dictOfReponses = {}
        reponse = listOfReponse[i][j]
        dictOfReponses["texte"] = reponse
        if '~' in reponse:
            dictOfReponses["correct"] = falseCorrect
        else:
            dictOfReponses["correct"] = trueCorrect

        listOfReponses.append(dictOfReponses)

    #Pour chaque question, on ajouter une question, une phrase et des ensembles de réponses dans un dictionnaire.
    dictOfQuiz["question"] = question
    dictOfQuiz["phrase"] = phrase
    dictOfQuiz["reponses"] = listOfReponses

    #Ajouter chaque dictionnaire dans une liste de Quiz
    listOfQuiz.append(dictOfQuiz)

# print(listOfQuiz)

#convertir une liste en JSON
quizJson = json.dumps(listOfQuiz)

# créer un fichier pour enregistrer le résultat
with open('quiz.json', 'w') as f:
    f.write(quizJson)

fichier.close()

```

Figure 53: Capture l'écran en illustrant de code Python (convertir file texte en JSON)

## 5. Arbre syntaxique

L'analyse syntaxique peut parfois être difficile pour les étudiants qui ne savent pas comment faire. Cependant, l'enseignement de la syntaxe a pour but de montrer que la syntaxe suit des règles régulières. L'enseignant peut par exemple choisir une méthode comme un graphe de dépendances pour expliquer les relations de dépendance entre les mots ou les groupes de mots.

Dans notre mémoire nous avons utilisé le graphe de dépendances dans la partie consacrée à l'exercice sur les parties du discours. Nous avons décidé d'intégrer le graphe de dépendances à cet exercice pour plusieurs raisons. Tout d'abord, les graphes de dépendances fournissent une représentation visuelle de la structure grammaticale d'une phrase, ce qui facilite la compréhension de la manière dont les éléments se rapportent les uns aux autres <sup>34</sup>. Ensuite, cela facilite l'analyse grammaticale en identifiant les relations entre les mots, permettant ainsi une analyse plus approfondie et précise des propriétés linguistiques<sup>35</sup>. Enfin, les graphes de

34 Li, X., & Zheng, Y. (2018). *Dependency Parsing Using a Neural Network for Syntactic and Semantic Analysis.*, 360-370.

dépendances offrent une manière visuelle et structurée d'enseigner la syntaxe, aidant les apprenants à comprendre les concepts syntaxes de manière concrète.

Comment évoqué lors de la présentation d'Universal Dependencies , le projet met à disposition l'outil Arborator qui permet de convertir des données qui sont au format CoNLL en un graphique (arbre syntaxique).

Pour la première version de notre application, nous avons choisi cet outil pour générer un graphe de dépendances. Voici le résultat qu'on a obtenu :

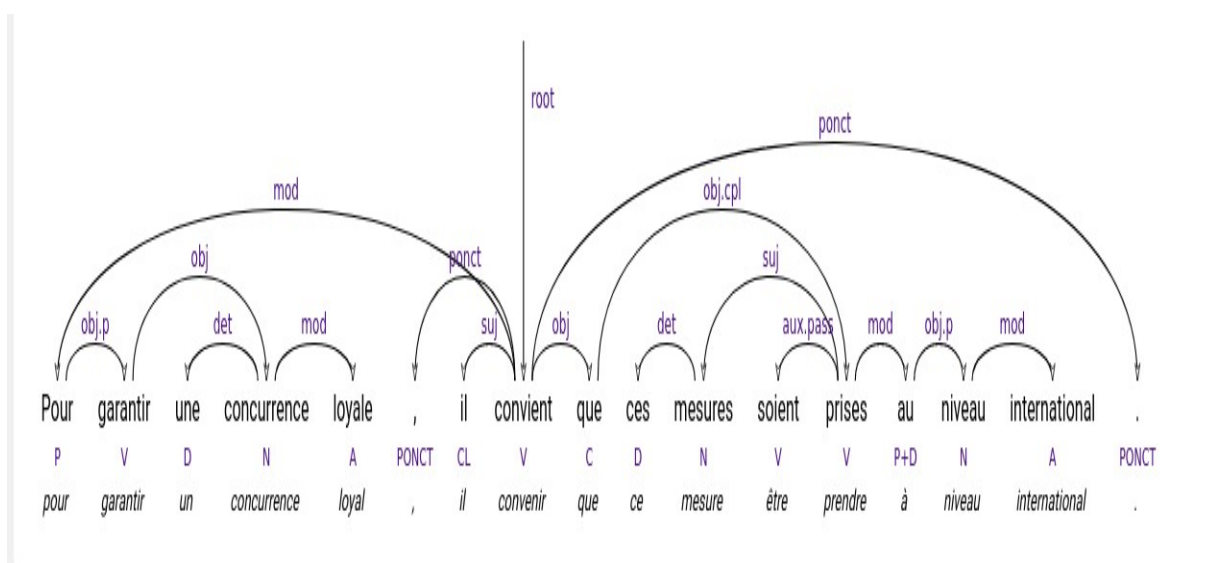


Figure 54: Une capture d'écran d'un graphe dépendance sur notre site

## 6. Préparation des données pour l'exercice sur la fonction grammaticale (fichier QuizCODCOI)

Afin d'automatiser la génération d'exercices visant à identifier la fonction grammaticale d'un mot au sein d'une phrase, nous avons opté pour l'utilisation de la bibliothèque Stanza car comme vu précédemment dans le comparatif des différents outils, elle nous produisait des résultats satisfaisants.

Le processus d'annotation des corpus avec Stanza débute en construisant un Pipeline (`nlp = stanza.Pipeline(lang='fr',processors='tokenize,pos,lemma,ner')`). Le pipeline effectue la

35 Christopher D. Manning, Hinrich & Schütze, H. (1999). *Foundations of Statistical Natural Language Processing*.

tokenisation, l'étiquetage grammatical (POS), la lemmatisation, les étiquettes POS (Part of Speech) et les relations de dépendance entre les mots (DEPREL).

Afin de traiter plusieurs phrases avec Stanza, nous avons créé un objet **corpus** qui contient les phrases sous forme d'une liste de chaîne de caractère (*string*). Ensuite, pour parcourir chacune des phrases de l'objet **corpus**, nous utilisons la boucle « for ». Chaque phrase est analysée par le modèle de traitement de langage naturel (pipeline), et on stocke ensuite les résultats détaillés de l'analyse textuelle effectuée par le modèle pipeline dans un objet **doc**. Cet objet **doc** contient toutes les informations résultant de l'analyse du texte, comme les tokens (mots), leurs propriétés grammaticales et sémantiques, les entités nommées, les dépendances syntaxiques, etc.

```
[
  [
    {
      "id": 1,
      "text": "J'",
      "lemma": "il",
      "upos": "PRON",
      "feats": "Number=Sing|Person=1|PronType=Prs",
      "head": 2,
      "deprel": "nsubj",
      "start_char": 0,
      "end_char": 2,
      "ner": "0",
      "multi_ner": [
        "0"
      ]
    }
  ],
  {
    "id": 2,
    "text": "appelle",
    "lemma": "appeler",
    "upos": "VERB",
    "feats": "Mood=Ind|Number=Sing|Person=1|Tense=Pres|VerbForm=Fin",
    "head": 0,
    "deprel": "root",
    "start_char": 2,
    "end_char": 9,
    "ner": "0",
    "multi_ner": [
      "0"
    ]
  }
],
  {
    "id": 3,
    "text": "Clément",
    "lemma": "Clément",
    "upos": "PROPN",
    "head": 2,
    "deprel": "obj",
    "start_char": 10,
    "end_char": 17,
    "ner": "S-PER",
    "multi_ner": [
      "S-PER"
    ]
  }
],
]
```

Figure 55: Capture d'écran illustrant les résultats de l'analyse réalisée par le modèle Pipeline et stockés dans l'objet Doc.

Ensuite, on a créé un objet **sentence** qui permet de se référer à une phrase individuelle qui a été analysée et annotée par le modèle *Pipeline*. Cet objet **sentence** est utilisé pour accéder au token et aussi à l'étiquette de relation de dépendance (DEPREL). Le token et l'étiquette DEPREL nous permettent de créer une liste de phrases comme (*Il s' **\*\*en\*\*** souvient.*) et une liste de réponse avec un « flag » pour chacune d'entre elle, qui permet d'identifier la bonne réponse (correct : false ou correct : true). Pour définir les bonnes/mauvaises réponses, on parcourt chaque réponse possible (COD, COI, complément de lieu ...) que l'on compare avec la valeur de l'étiquette DEPREL.

Enfin, pour chaque phrase, on vérifie si le rôle de dépendance (DEPREL) du mot est égal à « obj »(COD) ou « iobj » (COI). Si c'est le cas, On entoure le mot de deux astérisques \*\*, et il deviendra donc le mot que les étudiants doivent identifier.

La liste des choix de réponses possible comprend : complément de lieu, complément du nom, complément de l'adjectif, complément d'agent, COD et COI. Dans cette liste, nous avons inclus COD, COI et complément de lieu, car certains pronoms tels que « y » et « en » peuvent jouer le rôle de COD, COI et de complément de lieu en fonction de la phrase. Voici quelques exemples explicatifs :

- Le pronom « y » peut remplacer un complément d'objet indirect ou un complément de lieu, selon le contexte de la phrase. Par exemple : « *Nous avons visité Paris l'été dernier. C'était la première fois que nous y allions.* » Dans cette phrase, le pronom « y » remplace « à Paris », qui est un complément de lieu. Le pronom « y » peut également remplacer un complément d'objet indirect. Par exemple : « *Tu penses à ton travail ? Oui, j'y pense.* » Ici, « y » remplace « à mon travail », qui est un COI du verbe « penser ».
- Le pronom « en » peut être utilisé pour remplacer soit un complément d'objet indirect (« *Tu as parlé de tes problèmes ? Oui, j'en ai parlé.* »), soit à un complément d'objet direct (« *Vous avez réservé trois places ? Oui, nous en avons réservé trois.* »), soit à un complément de lieu (« *Tu reviens de la piscine ? Oui, j'en reviens.* »).

Après avoir établi une liste possible de réponses pour notre exercice, nous avons décidé de rendre aléatoire l'ordre des choix de réponses. Cela permet d'éviter que l'étudiant ne réponde que grâce à sa mémoire s'il fait l'exercice à plusieurs reprises.

Enfin, à l'aide de Python, nous enregistrons les résultats au format JSON, incluant une liste de consignes, une liste de phrases, ainsi qu'une liste de réponses. Cette démarche nous permettra de les intégrer dans le code JavaScript.

## 7. Correction pour la partie fonction grammaticale (fichier CorrectionCOD\_COI)

Il est parfois difficile d'analyser la fonction des compléments dans une phrase. *Ce complément est-il COD ? Ce complément est-il COI ? Comment trouver le COD ou le COI dans une phrase sans se tromper ?* Ce sont des questions que les étudiants étrangers posent souvent à leurs enseignants. Afin de répondre à ces questions, il est important de maîtriser l'analyse syntaxique des phrases. Pour ce faire, être capables d'identifier le sujet, le verbe et le complément. Une fois la phrase segmentée, la stratégie d'identification est la suivante pour les fonctions SUJ, COD et COI :

- La substitution pronominale (3<sup>e</sup> pers.)
- La transformation passive
- La transformation clivée

Afin d'aider les étudiants à comprendre leurs erreurs sur cette partie, nous avons mis en place une correction, visible une fois l'exercice terminé. Cette correction est présentée sous forme de tableau où nous avons inclus la phrase, une question et la découpe de la phrase en sujet, verbe et complément avec des couleurs différentes. Cette correction donne non seulement la réponse, mais elle aide aussi les étudiants à analyser et comprendre plus en détail la syntaxe.

Pour faire cette correction, la méthode est la même que pour la partie **Préparation des données la fonction grammaticale**, nous avons employé le modèle Pipeline pour analyser chaque phrase, et stocker les résultats dans l'objet **doc**. De plus, nous avons utilisé l'objet **sentence** pour extraire chaque token (un mot) et aussi leur étiquette **DEPREL**. Le résultat final est sauvegardé au format JSON, ce qui permet de l'exploiter facilement en javascript. Voici une capture d'écran de la partie dédiée à la correction des COD et COI :

### COMPLÉMENT D'OBJET DIRECT (COD):

Phrase	Sujet	Verbe	COD
J'appelle Clément. => Je l'appelle.	J'	appelle	Clément
J'aime la musique. => Je l'aime.	J'	aime	musique
Je mange un sandwich. => Je le mange.	Je	mange	le
Je connais Cécile. => Je la connais.	Je	connais	la
Il prend les enfants dans ses bras => Ils les prend dans ses bras.	Il	prend	les
Je regarde moi-même dans le reflet de l'eau. => Je me regarde dans le reflet de l'eau.	Je	regarde	me

Figure 56: Une capture d'écran illustrant la section de correction COD

### COMPLÉMENT D'OBJET INDIRECT (COI):

Phrase	Sujet	COI	Verbe
Pierre parle à son ami. => Pierre lui parle.	Pierre	lui	parle
C'est de cela que je parle. => J'en parle.	J'	en	parle
Marie ressemble à toi. => Elle te ressemble.	Elle	te	ressemble
C'est à cela que je pense. => J'y pense.	J'	y	pense
C'est de cela que je me moque. => Je m'en moque.	J'	en	moque
C'est de cela qu'il se souvient => Il s'en souvient.	Il	en	souvient

Figure 57: Une capture d'écran illustrant la section de correction COI

## 8. Choix du format JSON

L'utilisation du format JSON présente de nombreux avantages. Avec JavaScript, nous pouvons facilement analyser, lire et manipuler les données JSON sans avoir besoin de bibliothèques supplémentaires. Cela rend le processus de récupération et de traitement des données pour le Quiz beaucoup plus simple et rapide. De plus, JSON utilise une syntaxe claire et facile à comprendre, basée sur des paires clé-valeur, ce qui permet d'organiser les questions et réponses du Quiz de manière hiérarchique et logique. Cela facilite la création et la manipulation de données, ce qui est essentiel pour un Quiz interactif et personnalisé.

Cette structure organisée de notre quiz permet de gérer facilement les interactions entre l'utilisateur et le Quiz, tout en facilitant la validation des réponses.

```
var quiz = [{
  question: "Donner la partie du discours du mot **garantir** dans la phrase:",
  phrase: "Pour **garantir** une concurrence loyale , il convient que ces mesures soient prises au niveau international . ",
  reponses: [{
    texte: "ADV_Int",
    correct: false
  }, {
    texte: "ADV",
    correct: false
  }, {
    texte: "ADJ_Excl/Int",
    correct: false
  }, {
    texte: "V_Inf",
    correct: true
  }, {
    texte: "ADJ",
    correct: false
  }, {
    texte: "Conj_de_Coord",
    correct: false
  }
  ]
}, {
  question: "Donner la partie du discours du mot **loyale** dans la phrase:",
  phrase: "Pour garantir une concurrence **loyale** , il convient que ces mesures soient prises au niveau international . ",
  reponses: [{
    texte: "DET",
    correct: false
  }, {
    texte: "PROQ_Int",
    correct: false
  }, {
    texte: "ADJ",
    correct: true
  }, {
    texte: "V_Part_Pr\u00e9s",
    correct: false
  }, {
    texte: "NC",
    correct: false
  }, {
    texte: "DET_Int",
    correct: false
  }
  ]
}, {
```

Figure 58: Un exemple de données stockées dans notre mémoire au format JSON.

## **9. Création de l'application Web**

### **9.1 Choix techniques**

Avec JavaScript, HTML et CSS, nous pouvons facilement manipuler les données JSON pour créer un jeu de quiz interactif, comme : afficher les questions, les phrases et les options de réponse. Ces trois langages nous aident à fabriquer un site web de Quiz interactif attrayant. HTML structure le contenu, CSS stylise l'apparence et Javascript ajoute l'aspect interactif. Grâce à cette combinaison, les utilisateurs peuvent répondre aux questions, suivre leur progression à travers leurs réponses (grâce à la bibliothèque chart.js de Javascript, nous pouvons créer des graphiques pour comparer le temps de réponse et le nombre d'essais entre les questions), recevoir des retours pour identifier les questions à réviser. De plus, ces trois langages nous permettent de mettre en place des paramètres d'accessibilité qui permettent au site web d'être utilisable par tous.

Enfin, pendant mon Master, j'ai eu la chance de suivre des cours sur Javascript, HTML et CSS. Ce projet m'a donc permis de mettre en pratique ce que j'avais appris en classe, et aussi d'améliorer mes compétences dans le domaine du développement web.

### **9.2 Les fonctions essentielles de l'application de quiz**

#### **9.2.1 L'ordre aléatoire des questions**

Dans notre quiz, nous avons choisi de présenter les questions du test dans un ordre aléatoire. Autrement dit, les questions seront différentes d'un étudiant à l'autre. Si un étudiant peut faire le test plusieurs fois, chaque tentative présentera aussi un ordre de questions/réponses différent.

Nous avons choisi de mélanger aléatoirement l'ordre des questions de chaque test pour plusieurs raisons. D'abord, cela empêche simplement les étudiants de mémoriser les réponses sans vraiment comprendre le contenu (Mayer, R. E., & Moreno, R. (2003)<sup>36</sup>. Ensuite, avoir les questions du test dans un ordre aléatoire nous permet d'encourager les étudiants à se concentrer durant le test et à éviter les distractions (Fredricks, J. A., Blumenfeld, P. C., & Paris, A. H. (2004)<sup>37</sup>.

---

36 Mayer, R. E., & Moreno, R. (2003). *Nine ways to reduce cognitive load in multimedia learning.*, p.43-52.

37 Fredricks, J. A., Blumenfeld, P. C., & Paris, A. H. (2004). *School engagement: Potential of the concept, state of the evidence.*, p.59-109.



### **9.2.2 Le compte à rebours**

De nombreuses recherches montrent que pour donner de la motivation aux « joueurs », il est important de leur donner la sensation de pouvoir gagner, et, même si ici les questions restent les mêmes, d'avoir une sensation de progression.

C'est pour cela que nous avons mis en place un compte à rebours qui nous aide à limiter le temps de réponse accordé aux étudiants. Cela rend le jeu plus difficile et passionnant.

Nous avons pour le moment décidé de donner un total de 5 minutes pour répondre à 15 questions, on attend donc en moyenne une réponse toutes les 20 secondes.

Nous pensons que ce délai est équilibré, ni trop court ni trop long, cela permet aux étudiants de répondre à toutes les questions, avec pour chaque question le droit à 3 tentatives.

On peut envisager dans le futur donner aux utilisateurs la possibilité de choisir ces paramètres (le temps limite accordé pour répondre, ainsi que le nombre maximum de tentatives).

### **9.2.3 Limiter le nombre de tentatives.**

Dans le but d'encourager les étudiants à engager une réflexion approfondie face aux questions nous avons décidé d'accorder trois tentatives pour chaque question. Cette stratégie offre plusieurs avantages par rapport à un quiz sur H5P. Sur H5P, les utilisateurs peuvent essayer toutes les questions, mais ils ne peuvent pas réessayer une question plusieurs fois. En permettant aux utilisateurs de tenter plusieurs fois une question, nous favorisons l'apprentissage actif et le développement de compétences analytiques. De plus, les étudiants peuvent réfléchir plus attentivement et sélectionner les bonnes réponses en apprenant à analyser chaque question de manière optimale. Le fait de pouvoir essayer plusieurs fois pour chaque question permet aux étudiants d'apprendre de leurs erreurs. Cela les aide à mémoriser les concepts en identifiant pourquoi ils ont fait des erreurs et en ajustant leurs réponses en conséquence.

Par exemple, imaginons que l'étudiant A doit trouver quelle est la fonction grammaticale du mot « en » dans la phrase « *Je m'en moque.* ». L'étudiant A sait que le pronom « en » peut être remplacé par COD, un complément de lieu, complément circonstanciel de cause ou un COI. Donc, il commence en essayant de penser à un objet direct. Cette réponse n'est pas correcte, alors il réfléchit à choisir entre le complément de lieu, le complément circonstanciel de cause et le COI. Il se demande si le mot « en » peut remplacer un complément de lieu, mais en regardant bien la phrase, il comprend que ce n'est pas possible. Ensuite, il pense au

complément de cause, il essaie cette réponse, mais cela ne semble pas bon non plus. Il lui reste un dernier choix, c'est le COI. L'étudiant A peut se poser la question « se moquer de quoi ? » et finalement, il choisit la bonne réponse, qui est un objet indirect (COI).

En conclusion, l'étudiant A doit réfléchir et analyser la question ainsi que les réponses pour trouver la bonne réponse. De plus, même s'il choisit une mauvaise réponse deux fois de suite, il peut quand même comprendre son erreur et éviter de la refaire pour d'autres questions similaires.

#### **9.2.4 Barre de progression**

Afin d'indiquer aux étudiants l'état d'avancement de leurs tests, nous avons intégré une barre de progression. Cette idée a été inspirée de la barre de bonnes réponses dans le jeu Quiz sur H5P. Cependant, nous avons apporté des améliorations significatives. Contrairement à H5P, où chaque réponse possède sa propre barre de progression distincte, notre approche intègre une seule barre de progression globale. De plus, contrairement à H5P, où la barre de progression n'est visible qu'une fois que l'utilisateur a répondu à une question, notre barre de progression est visible pendant qu'il répond aux questions. Cette amélioration ou du moins ce fonctionnement différent permet aux utilisateurs de suivre leur avancement tout au long du test, ce qui favorise une expérience plus claire et intéressante.

Lorsqu'un utilisateur répond correctement à une question, la barre de progression avance en vert et affiche le message « Bravo ». Cette approche encourage les utilisateurs à trouver les bonnes réponses et les motive à continuer de réussir.

Question 1/15 09:52 Paramètres d'accessibilité

**Donner la partie du discours du mot **convient** dans la phrase:**  
 Pour garantir une concurrence loyale , il **convient** que ces mesures soient prises au niveau international .

☐ Conj\_de\_Sub  
☒ V  
☐ PRO\_Int  
☐ Mot\_étranger  
☐ Conj\_de\_Coord  
☐ DET

☐ Voir la syntaxe de la phrase  
 Bravo!

Next

Figure 59: Capture d'écran de la barre de progression en vert.

Cependant, si l'utilisateur ne répond pas correctement, la barre de progression ne progresse pas et change de couleur pour devenir rouge. Cela signifie que l'utilisateur a raté et doit faire plus d'efforts pour retrouver la barre de succès en vert. Nous avons ajouté cet élément pour créer une expérience de jeu réaliste. Si l'utilisateur gagne, il obtient quelque chose, s'il ne gagne pas, il perd quelque chose. Cela le motive à continuer de jouer pour gagner encore plus.

Question 2/15 07:41 Paramètres d'accessibilité

**Donner la partie du discours du mot **au** dans la phrase:**  
 Pour garantir une concurrence loyale , il convient que ces mesures soient prises **au** niveau international .

☐ V\_Imp  
☒ V\_Inf  
☐ ADV  
☐ PREP\_+\_DET  
☐ Conj\_de\_Sub  
☐ Conj\_de\_Coord

☐ Voir la syntaxe de la phrase  
 Vous avez encore 2 essais !


Valider

Figure 60: Capture d'écran de la barre de progression en rouge.

Question 3/15 06:07 [Paramètres d'accessibilité](#)

**Donner la partie du discours du mot **prises** dans la phrase:**  
*Pour garantir une concurrence loyale , il convient que ces mesures soient **prises** au niveau international .*

☐ Interj  
☐ Préf  
☐ ADV\_Int  
☐ PREP+\_DET  
☐ DET\_Int  
☒ V\_Part\_Passé



[Voir la syntaxe de la phrase](#)

Bravo!

[Next](#)

Figure 61: Capture d'écran montrant la barre de progression qui change de couleur en vert et avance.

### 9.2.5 Choix de bouton

Étant donné que l'utilisateur ne doit choisir qu'une et une seule bonne réponse, nous avons choisi de mettre des boutons radio qui correspondent parfaitement à ce besoin. En voyant ces boutons l'utilisateur sait qu'il ne doit donner qu'une réponse, ce qui n'est pas le cas si nous mettons par exemple des cases à cocher (checkbox), qui pourraient l'amener à se demander s'il s'agit d'un QCM (questionnaire à choix multiple) ou d'un QCU (questionnaire à choix unique).

De plus, étant donné que nous avons tout de même 6 réponses, nous avons choisi de disposer les choix de réponses verticalement, afin que ce soit le plus clair et visible possible pour l'utilisateur.

### 9.3 Le graphique présentant les réponses des étudiants

Sur Moodle, nous avons la possibilité de consulter les résultats d'exercices de chaque élève dans les quiz, classés par date, score minimum, score maximum et durée. Cependant, il n'est pas possible d'enregistrer le temps et le nombre d'essais pour chacune des réponses, nous n'avons que des statistiques globales. Pour résoudre ce problème, on propose d'ajouter un diagramme supplémentaire à côté du Quiz. Pour créer le graphique, on a utilisé la librairie

Chart.js. Cette bibliothèque (gratuite et open-source) est populaire et mise à jour régulièrement.

La fonction principale de ce diagramme est de comparer le temps que les utilisateurs prennent pour répondre à chaque question, et combien de tentatives ils ont effectuées. Cela permettra aux étudiants de visualiser rapidement leurs progrès.

Afin de rendre les choses claires, nous avons choisi d'utiliser un diagramme en bâton empilés. Ces diagrammes sont très efficace pour la comparaison de données, de plus ils sont faciles à lire.

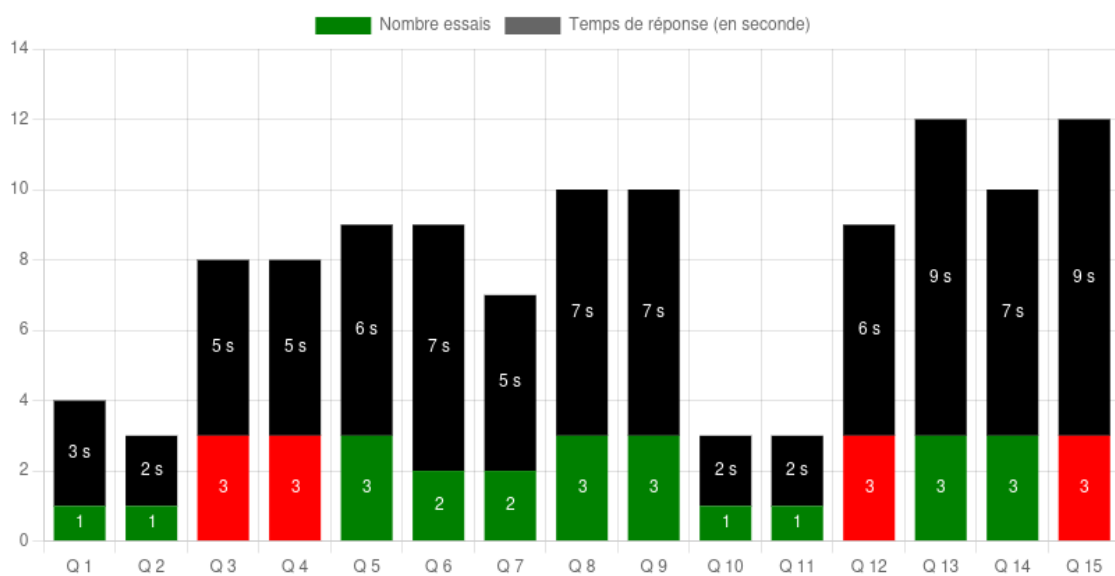
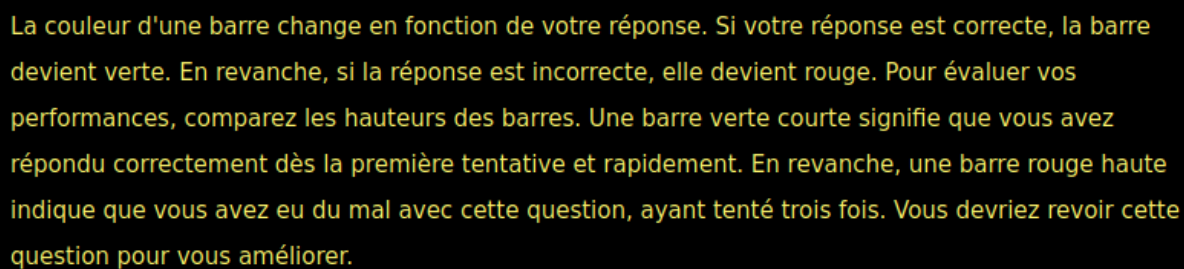


Figure 62: Une capture d'écran présentant le résultat de l'exercice au sein du projet, incluant le graphique empilé.

Dans cet exemple, on observe que l'étudiant A répond rapidement et correctement aux questions 1, 2, 10 et 11, étant donné que les barres correspondantes sont vertes et moins hautes que les autres. Ainsi, pour l'étudiant A, ces quatre questions semblent faciles. Par contre, les questions 3, 4, 12 et 15 posent des difficultés, car les barres associées sont rouges et plus élevées que les autres. Cela indique des problèmes pour trouver les réponses. L'étudiant A a tenté trois fois, investissant plus de temps que pour les autres questions, sans parvenir à la bonne réponse. Par conséquent, il est essentiel que l'étudiant A se penche davantage sur ces questions et les révise soigneusement. C'est également le cas des questions

5, 8, 9, 13, 14, où l'étudiant A a également pris plus de temps et a effectué trois essais pour parvenir à la bonne réponse. Il est donc recommandé de revoir ces questions également.

1. **Un guide pour comprendre le diagramme** : Pour aider les étudiants à interpréter les résultats du diagramme, on a inclus un paragraphe explicatif. Ce paragraphe clarifie les significations des couleurs de fond des barres. Par exemple, la couleur verte indique une réponse correcte, tandis que la couleur rouge signale une réponse incorrecte. De plus, nous avons abordé la signification de la hauteur des barres. Dans ce contexte, la hauteur des barres correspond au nombre total d'essais effectués et du temps passé. Si une barre est élevée, cela indique que les questions associées nécessitent une révision approfondie. En revanche, si une barre est basse, cela suggère que les questions sont relativement simples pour l'étudiant.



La couleur d'une barre change en fonction de votre réponse. Si votre réponse est correcte, la barre devient verte. En revanche, si la réponse est incorrecte, elle devient rouge. Pour évaluer vos performances, comparez les hauteurs des barres. Une barre verte courte signifie que vous avez répondu correctement dès la première tentative et rapidement. En revanche, une barre rouge haute indique que vous avez eu du mal avec cette question, ayant tenté trois fois. Vous devriez revoir cette question pour vous améliorer.

Figure 63: Une capture d'écran illustrant le guide d'interprétation du diagramme au sein de notre projet.

2. **Feedback sur les réponses** : Lorsque les enseignants examinent les copies des étudiants, ils fournissent souvent des commentaires pour aider les étudiants à comprendre et identifier leurs lacunes. Cependant, étant donné que l'objectif de ce mémoire est de créer des exercices de manière automatique, cela signifie que les enseignants ne peuvent pas consacrer de temps à corriger chaque réponse individuellement ni à fournir des commentaires personnalisés. C'est pourquoi nous avons eu l'idée d'intégrer des commentaires de manière automatisée. Pour ce faire, nous avons adopté une approche simple : si une réponse est incorrecte, cela suggère que les étudiants doivent travailler davantage sur cette question. En cas de réponse correcte, mais avec plus de 2 tentatives ou si le temps passé pour une question dépasse les 20 secondes, cela indique que l'élément de syntaxe sur lequel portait la question n'est probablement pas maîtrisé.

Il faut réviser la question **3** pour retrouver la partie du discours de mot **que**.

Il faut réviser la question **4** pour retrouver la partie du discours de mot **il**.

Vous avez trouvé une bonne réponse, mais il faut encore s'entraîner sur la question **5** pour répondre plus rapidement la prochaine fois.

Vous avez trouvé une bonne réponse, mais il faut encore s'entraîner sur la question **8** pour répondre plus rapidement la prochaine fois.

Vous avez trouvé une bonne réponse, mais il faut encore s'entraîner sur la question **9** pour répondre plus rapidement la prochaine fois.

Il faut réviser la question **12** pour retrouver la partie du discours de mot **autres**.

Vous avez trouvé une bonne réponse, mais il faut encore s'entraîner sur la question **13** pour répondre plus rapidement la prochaine fois.

Vous avez trouvé une bonne réponse, mais il faut encore s'entraîner sur la question **14** pour répondre plus rapidement la prochaine fois.

Il faut réviser la question **15** pour retrouver la partie du discours de mot **acoustiques**.

Figure 64: Une capture d'écran illustrant le Feedback sur les réponse au sein de notre projet.

En résumé, le graphique fournit un suivi approfondi de la progression de l'étudiant en présentant des informations importantes comme la durée et le nombre d'essais. Ces données constituent une base solide pour fournir un retour d'information précis à l'étudiant. Cela permet de mettre en évidence les domaines nécessitant des améliorations et de prodiguer des conseils spécifiques pour stimuler sa progression. Notre objectif est que grâce à ces informations précises, l'étudiant puisse mieux évaluer sa performance et travailler de manière plus ciblée pour atteindre ses objectifs d'apprentissage.

#### 9.4 L'accessibilité d'un site WEB

Selon deux études publiées le jeudi 2 février 2023 par le service statistique des ministères sociaux, on estime que 7,7 millions de personnes âgées de plus de 15 ans sont en situation de handicap en France. Parmi elles, 40 000 étudiants, selon le site [handicap.fr](http://handicap.fr), enregistré à la rentrée 2021. Le site [aveuglesdefrance](http://aveuglesdefrance), rapporte que près de 1,7 million de personnes en France sont atteintes d'un trouble de la vision.

Malheureusement, l'accessibilité numérique pour les personnes en situation de handicap reste un défi. Selon un article de France Culture publié le mercredi 15 juin 2022, [En France, une infime partie des sites Internet est accessible aux personnes en situation de handicap](#), moins de 10 % des sites Internet sont réellement accessibles aux personnes handicapées. Cette situation

empêche spécialement les personnes handicapées visuelles d'accéder à de nombreux sites pour lire des informations ou effectuer des recherches sur Internet.

Afin d'aider les personnes handicapées à accéder au monde numérique, il est essentiel de s'engager activement dans l'accessibilité numérique. Cela signifie rendre les sites Internet plus conviviaux et adaptés aux différents besoins des personnes handicapées.

Comme mentionné dans notre introduction, les jeux de Quiz sur Moodle ou H5P souffrent d'un manque d'accessibilité. Cela signifie que les étudiants ayant des problèmes de vue par exemple ne peuvent pas effectuer (ou pas confortablement) les exercices sur ces plateformes. C'est pourquoi, dans notre projet, nous avons considéré l'accessibilité du site web comme une de nos priorités. Cependant, en raison du temps limité pour travailler sur ce projet, nous nous sommes concentré principalement sur les étudiants malvoyants (dans l'idéal il faudrait implémenter de l'audio pour les utilisateurs non voyants par exemple).

Pour rendre notre site personnalisable par les utilisateurs selon leurs préférences/besoins, nous avons ajouté un bouton qui permet de déployer un « panneau » qui présente les paramètres d'accessibilité. Ce panneau permet aux utilisateurs de personnaliser leur expérience en choisissant la couleur de fond et la couleur du texte parmi 40 options différentes. Ils peuvent également sélectionner la police du texte, ajuster la taille du texte, l'espace entre les lignes et l'alignement du texte (à droite ou à gauche) selon leurs préférences. De plus, nous avons ajouté un mode jour et nuit pour offrir une expérience visuelle adaptée à différentes conditions d'éclairage.

Après avoir ajusté les paramètres selon leurs préférences, les utilisateurs peuvent simplement cacher ces options en cliquant sur le bouton « Cacher les paramètres d'accessibilité ». Cela leur permet de se focaliser uniquement sur la partie Quiz du site.



Figure 65: Une capture d'écran des paramètres d'accessibilité disponibles sur notre site.



#### 9.4.1 La police du texte

Selon le Ministère de l'Éducation nationale dans le livre « *Accessibilité et Adaptabilité des Ressources Numériques pour l'École* » (Mars 2018), il est recommandé d'utiliser des polices de caractères simples et sans empattement. Ces polices rendent la lecture plus facile pour les parties de texte isolées, comme les menus, les étiquettes des messages ou les champs de saisie. De plus, le livre « *Le guide de l'accessibilité numérique* » de Contentsquare souligne que les polices bâtons ou sans-serif sont simples et offrent un espacement régulier, ce qui les rend accessibles aux individus atteints de troubles cognitifs.

Dans l'article « *Good fonts for dyslexia* » de Luz Rello<sup>38</sup> et Ricardo Baeza-Yates<sup>39</sup>, une expérience a été réalisée avec 48 participants dyslexiques (22 femmes, 26 hommes) qui devaient lire 12 textes comparables, chacun lit dans un type de police différent. Deux critères ont été évalués : le temps de lecture et la durée de fixation.

Les résultats ont montré que la police *Arial It* a pris plus de temps à lire que les polices *Helvetica*, *Arial* et que les yeux se sont arrêtés plus longtemps sur la police *Arial It* que sur la plupart des autres polices. En résumé, les polices sans empattement (comme *Verdana*, *Arial*, *Helvetica*), à largeur fixe et en caractères à bâton ont grandement amélioré la vitesse de lecture par rapport aux polices avec empattement (comme *Sérial* ou *Time*).

#### 9.4.2 Taille du texte

Il est essentiel de choisir la bonne taille de texte pour que notre site soit accessible aux personnes malvoyantes. Pour cela, nous recommandons d'utiliser une taille de police d'au moins 14 pixels, comme indiqué dans « *Le guide de l'accessibilité numérique* » de Contentsquare et « *Accessibilité et Adaptabilité des Ressources Numériques pour l'École* » du Ministère de l'Éducation nationale.

Sur notre site, on a ajouté deux boutons pour aider les utilisateurs : un bouton « plus » pour augmenter la taille du texte selon leurs préférences, sans aucune limite, et un autre bouton « moins » pour diminuer la taille du texte. Cela permet à chacun d'ajuster la taille du texte à ses préférences personnelles, pour une lecture confortable et adaptée à ses besoins visuels.

---

38 Luz Rello: Fondateur de Change Dyslexia ([www.changedyslexia.org](http://www.changedyslexia.org)), une organisation sociale dédiée au dépistage et au traitement de la dyslexie en utilisant Dyteactive, une plateforme de recherche pour la dyslexie.

39 Ricardo Baeza-Yates: Professeur du département of Information and Communication Technologies (DTIC) Barcelona, Spain



Figure 66: Une capture d'écran montrant l'affichage de la taille du texte avec deux boutons, l'un pour augmenter la taille (« plus ») et l'autre pour la diminuer (« moins »).

### 9.4.3 Alignement

Selon le site W3C (World Wide Web Consortium)<sup>40</sup>, il est recommandé de ne pas justifier le texte, c'est-à-dire de ne pas aligner le texte sur les marges gauche et droite <sup>41</sup>. Lorsque le texte est justifié, il peut présenter des espaces de largeur variable entre les caractères, ce qui peut rendre difficile pour l'utilisateur de distinguer clairement le début ou la fin d'un mot. Par conséquent, il est préférable d'aligner le texte uniquement sur les marges gauche, car cela crée un espacement cohérent entre les mots.

Sur notre site, on a créé deux boutons permettant aux utilisateurs de choisir l'alignement du texte, soit à gauche, soit à droite, en fonction de leurs besoins et préférences.

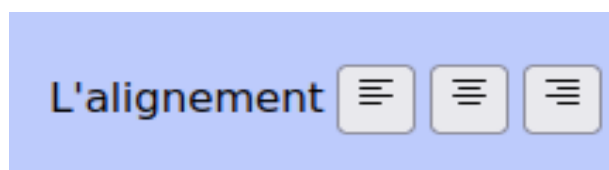


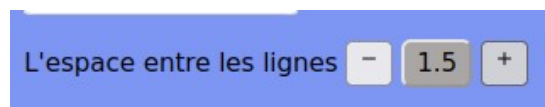
Figure 67: Une capture d'écran illustrant l'affichage des options d'alignement du texte avec deux boutons distincts : l'un pour l'alignement à gauche et l'autre pour l'alignement à droite

### 9.4.4 L'interligne

Selon le Web Content Accessibility Guidelines (WCAG) et les recommandations du Ministère de l'Éducation nationale, il est recommandé d'espacer le texte dans les paragraphes en utilisant un espace équivalent à au moins une ligne et demie. Cette astuce aide à rendre le texte plus facile à lire et à comprendre. Cela est particulièrement bénéfique pour les personnes ayant des problèmes de vue ou des besoins spéciaux d'accessibilité.

40 Web Content Accessibility Guidelines (WCAG) 2.1, <https://www.w3.org/TR/WCAG21/>

41 « Text is not justified (aligned to both the left and the right margins). », <https://www.w3.org/TR/WCAG21/>



*Figure 68: Une capture d'écran illustrant l'affichage de l'interligne*

## **VI – AXES D'AMÉLIORATION DU PROJET**

### **1. Point technique**

Dans la version «1.0 » du projet présentée dans ce mémoire, les utilisateurs ont la possibilité d'effectuer des exercices sans avoir à s'inscrire. Cependant, s'ils souhaitent consulter leurs résultats, par exemple demain ou le mois prochain, cette fonctionnalité n'est pas disponible.

Afin d'améliorer cette expérience, nous prévoyons d'introduire deux modes d'utilisation :

- Un mode invité : Il s'agit du fonctionnement actuel, pas besoin de se connecter pour pouvoir faire les exercices. Cependant aucune donnée, aucun historique n'est sauvegardé
- Un mode connecté : L'utilisateur crée un compte, puis se connecte à son compte pour faire les exercices. Cela nécessitera la mise en place d'une base de données, qui permettra de stocker les utilisateurs, garder un historique des exercices effectués par les utilisateurs, les scores obtenus, les temps de réponses etc etc

De plus, l'ajout d'une base de données et d'un mode connecté nous permettrait d'introduire la notion de profil (avec pour commencer, des profils enseignant ou étudiant).

On peut imaginer que les enseignants auraient accès à un tableau de bord, dans lequel ils pourraient consulter la liste des étudiants, et pour chacun de ces étudiants, un compte rendu du nombre d'exercices effectués, des graphiques de progression de l'étudiant... Si le professeur clique sur un exercice en particulier, il pourra voir pour chaque question le temps de réponse et le nombre d'essais, de manière similaire à ce que l'étudiant voit lorsqu'il effectue son exercice.

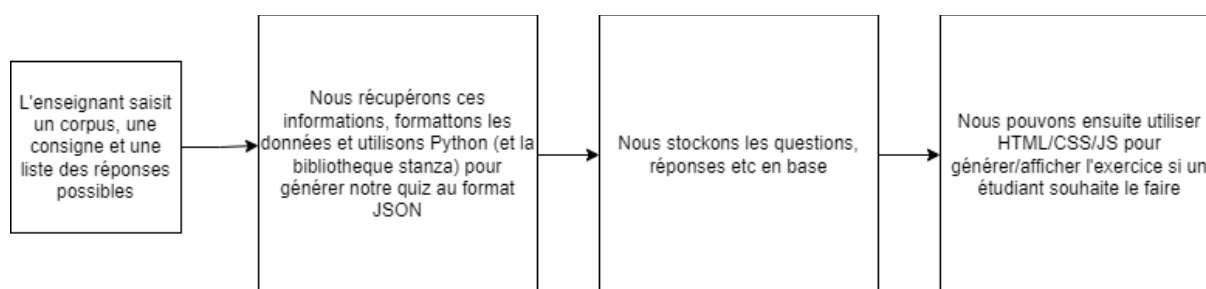
On pourrait même envisager d'utiliser le feedback que l'on a mis en place pour afficher les points forts / points faibles de l'étudiant, afin de pouvoir leur proposer des exercices appropriés.

Du côté des étudiants, une fois connectés, ils auront accès à leurs exercices et pourront consulter leurs résultats et leurs progression. Afin de stimuler les étudiants, on peut aussi imaginer mettre en place un système de points et de classements entre les étudiants. On pourrait par exemple accorder 3 points pour une bonne réponse au premier essai, 2 points au deuxième essai et 1 point au dernier essai. Avec en plus des bonus pour une série de bonnes réponses ... ce genre de chose. Le but serait que les étudiants voient la plateforme comme un jeu, afin de les encourager à revenir, réessayer etc ... Il ne fait aucun doute que c'est ce genre de fonctionnement qui attire beaucoup d'utilisateurs sur des applications comme DuoLingo (une application d'apprentissage des langues) par exemple. Ils ont notamment mis en place un système de « streak », qui encourage les utilisateurs à se connecter chaque jour, ne serait-ce que pour s'entraîner quelques minutes.

Une des fonctionnalités majeure qu'il faudrait mettre en place, pour répondre notamment à un des besoins principaux des professeurs qui est le gain de temps, serait de permettre à ceux-ci de créer, via leur propre corpus, des exercices automatisés. Nous avons montré qu'il est possible de générer ceux-ci automatiquement, notamment pour les exercices sur les fonctions grammaticales COD,COI, cependant nous avons utilisé les corpus de départ directement dans notre code.

L'idée serait donc de récupérer le corpus que le professeur saisira sur la plateforme, ainsi que la consigne pour l'exercice et les choix de réponses qu'il souhaite, et ensuite d'effectuer le traitement déjà en place pour arriver jusqu'à la génération des différentes questions/réponses du quiz. Nous aurons besoin pour cela d'utiliser un serveur web (apache par exemple), qui est de toute manière nécessaire pour rendre disponible notre application sur le web. De plus il faudrait faire une étude technique, notamment pour décider de la technologie à utiliser pour le backend (python avec le framework Django par exemple) ?

Le fonctionnement (sans rentrer dans le détail) serait le suivant :



Plutôt que d'utiliser Stanza, on pourrait par exemple utiliser l'[API UDPipe](#) pour analyser le corpus fourni par le professeur.

## 2. Point de linguistique

Dans notre projet, on a réussi à afficher un arbre syntaxique. Cependant, il faut utiliser le site <https://arboratorgrew.elizia.net/> pour générer la représentation visuelle de cet arbre syntaxique. Donc, nous sommes en train de réfléchir à une manière de générer automatiquement ces arbres syntaxiques directement sur notre plateforme. Si c'est réalisable, nous aimerions également offrir aux étudiants la possibilité de choisir le modèle d'arbre syntaxique qu'ils préfèrent.

On souhaite aussi de créer plusieurs types d'exercices de syntaxes différentes, et aussi des exercices de morphologies, de la sémantique, etc. pour que les étudiants puissent améliorer leurs compétences linguistique.

## 3. Créer d'autres activités que des Quiz

Notre projet propose pour le moment uniquement des quiz. A l'avenir, il serait intéressant d'intégrer, si possible de manière totalement automatique encore une fois, d'autres activités, comme les glisser-déposer, surlignage de mots, et bien d'autres encore.

## 4. Accessibilité de notre application

Enfin, il faudrait idéalement que notre application puisse être utilisée sur toutes les plateformes (mobile, tablette, pc ...). Il faudrait donc faire soit une application « responsive », soit une application web et une application mobile par exemple.

## VII - CONCLUSION

Un des objectifs principaux de ce mémoire était d'évaluer la possibilité de créer de manière automatique des exercices d'auto-évaluation portant sur la syntaxe à partir de corpus annotés. Pour cela nous avons évalué dans quelle mesure les outils UDPipe, Stanza, Spacy étaient capables de produire des corpus annotés de bonne qualité.

Nous avons pu constater que cela dépend des corpus et des outils. En effet, les résultats pour l'identification des compléments COD/COI étaient très bons pour Stanza et UDPipe, beaucoup moins pour Spacy. Pour le corpus contenant des attributs du COD, les résultats étaient très mauvais et inexploitable pour les trois outils, et donc non utilisables pour la création d'un exercice.

Même si nous avons donc pu utiliser un corpus annoté pour générer un exercice sur les compléments COD/COI de manière automatique, il est évident (au vu de l'analyse de notre second corpus notamment) qu'il reste beaucoup de travail pour pouvoir faire la même chose avec d'autres types d'exercices.

Pour permettre aux étudiants de s'entraîner sur nos exercices, nous avons décidé de créer notre propre site web plutôt que d'utiliser les plateformes existantes (Moodle par exemple). Plusieurs raisons à cela, tout d'abord nous souhaitons permettre à n'importe qui de pouvoir effectuer nos exercices, et cela sans inscription. De plus, nous n'étions pas satisfaits avec les paramètres d'accessibilité disponibles sur les sites existants. Nous avons donc pu mettre en place sur notre site divers paramètres personnalisables, afin de permettre aux personnes malvoyantes notamment de définir l'affichage qui leur convient le mieux.

De plus, créer notre propre site nous a permis d'ajouter des fonctionnalités que nous estimions importante, par exemple la possibilité d'avoir des informations sur le nombre d'essais et le temps pris pour chaque question, ou encore des paramètres d'accessibilité. Même si pour l'instant cette information n'est disponible qu'à l'utilisateur faisant le test, on peut imaginer dans le futur que les professeurs soient en capacité de consulter ces informations, et de conseiller les étudiants en fonction des résultats observés.

Malgré tout, il est évident qu'arriver à créer une plateforme aussi aboutie que Moodle par exemple demanderait énormément de travail et de moyens. Nous aimerions tout de même pousser notre projet et y ajouter de nombreuses fonctionnalités, dont la possibilité pour les professeurs de créer des exercices de manière automatique (portant sur les compléments COD/COI pour le moment), juste en fournissant un corpus qu'ils souhaitent exploiter.

Enfin, nous nous sommes concentré ici sur la création des QCU (questions à choix uniques), car pour être honnête il s'agissait du type d'exercice le plus simple à mettre en place de manière automatique. Il nous semble indispensable que d'autres types d'exercices soient disponibles, afin de garder intéressés les étudiants (par exemple des exercices de type *drag and drop* ou encore *mark the words*).

A l'heure actuelle, il paraît malgré tout difficile de pouvoir créer une plateforme qui permettrait de créer un grand panel d'exercices différents, et ce quasiment sans actions manuelle des enseignants.

Certes nous avons identifié certaines fonctionnalités qui pourraient être implémentées sur les plateformes en ligne, et pourquoi pas sur notre site web, et qui permettraient aux professeurs de gagner du temps (par exemple un feedback automatique sur les lacunes des étudiants en fonction de leurs résultats aux exercices, ou encore la possibilité de calculer des notes automatiquement etc ...).

Malgré tout, la partie création automatique de quiz à partir d'un corpus semble pour le moment présenter certaines limites. Nous avons de très bons outils à notre disposition pour annoter les corpus, cependant en fonction des corpus que l'on souhaite analyser, et plus précisément en fonction des éléments de syntaxe sur lesquels nous souhaitons tester les étudiants, les résultats sont très variables, et parfois peu ou pas fiables.

Il conviendrait donc de creuser la partie annotation de corpus, et de rechercher, tester voir pourquoi pas développer d'autres outils qui nous permettraient d'obtenir des annotations suffisamment précises pour générer un grand nombre d'exercices différents de manière automatique, sans pour autant prendre le risque de fournir aux étudiants des exercices/corrections erronées.

## BIBLIOGRAPHIE

A.Y. M. De-Souza & E. K. Bakah (2010). « Difficultés d'emploi des pronoms personnels COD et COI dans les écrits des étudiants du Département de Français à l'Université de Cape Coast »

Benjamin S. Bloom (1984). « *The 2 sigma problem: The search for methods of group instruction as effective as one-to-one tutoring.* », University of Chicago and Northwestern University, 4-16.

Caroline Lachet. (2014), « *Pratiques et représentations grammaticales des étudiants à l'université* », Université Paris Descartes / Sorbonne Paris Cité, France, Laboratoire EDA (Éducation, Discours, Apprentissages)

Christopher D. Manning, Hinrich & Schütze, H. (1999). « Foundations of Statistical Natural Language Processing. »

Christopher Manning, Richard Soche (2019), « Lecture Notes: Part IV, Dependency Parsing »

Fredricks, J. A., Blumenfeld, P. C., & Paris, A. H. (2004). « School engagement: Potential of the concept, state of the evidence. », p.59-109

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajič, Christopher Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. (2016), Universal Dependencies v1: A multilingual treebank collection. In *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC 2016)*. European Language Resources Association, Portorož, Slovenia, p.1659–1666

Jean-Adolphe Rondal (2017), « *Peut-on se dispenser des catégories grammaticales, des hiérarchies et des règles formelles dans l'acquisition de la syntaxe en langue naturelle ?* », L'Année Psychologique, (Vol. 117), p.1 à 40

Li, X., & Zheng, Y. (2018). Dependency Parsing Using a Neural Network for Syntactic and Semantic Analysis., 360-370.

Marie-Catherine de Marneffe and Christopher D. Manning, « Stanford typed dependencies manual », September 2008.



Mayer, R. E., & Moreno, R. (2003). « Nine ways to reduce cognitive load in multimedia learning. », p.43-52.

STRAKA M. & STRAKOVÁ J. (2017). « Tokenizing, POS Tagging, Lemmatizing and Parsing UD 2.0 with UDPipe », Vancouver, Canada

## SITE INTERNET

Antonio BALVET, « Projet ACE », <https://github.com/abalvet/ACE/blob/main/v0.9/moodle-quizzes/Sequoia-subset-quiz.gift>, consulté le 20 juin 2023

« Achèvement des activités », [https://docs.moodle.org/3x/fr/Achèvement\\_des\\_activités](https://docs.moodle.org/3x/fr/Achèvement_des_activités), juin 2023

« Catalogue », <https://docs.moodle.org/31/en/blocks/catalogue>, consulté le 15 juin 2023

« Carte interactive montrant où WIMS est utilisé », <http://downloadcenter.wimsedu.info/download/map/map2.html>, consulté le 15 juin 2023

« Des informations sur le format CoNLL-U Format », <https://universaldependencies.org/format.html>, consulté le 6 juillet 2023.

« Enseignement : la France dans le classement PISA », 2020, 21 août, <https://www.vie-publique.fr/eclairage/19539-resultats-des-eleves-la-france-et-le-classement-pisa#:~:text=La France compte respectivement 493,mathématiques en 551 en sciences> consulté le 18 mai 2023.

Éric Chaverou, 2022, 22 juin, « En France, une infime partie des sites Internet est accessible aux personnes en situation de handicap ». En ligne : <https://www.radiofrance.fr/franceculture/en-france-une-infime-partie-des-sites-internet-est-accessible-aux-personnes-en-situation-de-handicap-2872411>, consulté le 10, août, 2023.

« Fixez l'ordre des exercices de la série, Paramétrer l'exercice » , <https://wims.univ-cotedazur.fr/wims/wims.cgi?session=DN00FDA16F.2&+lang=en&+module=help%2Fteacher%2Fdocbeginner.fr&+cmd=reply&+job=read&+foldin=30..4#primi.4>, consulté le 15 juin 2023

Javascript <https://developer.mozilla.org/fr/docs/Web/JavaScript>, consulté le 20 juillet 2023.

H5P, <https://h5p.org/> consulté le 5 juin 2023.

« Learning management system »,

[https://fr.wikipedia.org/wiki/Learning\\_management\\_system](https://fr.wikipedia.org/wiki/Learning_management_system), consulté le 14 juin 2023

« Lesson activity », [https://docs.moodle.org/31/en/Lesson\\_activity](https://docs.moodle.org/31/en/Lesson_activity), consulté le 14 juin 2023

« Licence publique générale gnu », <http://www.gnu.org/copyleft/gpl.html>, consulté le 14 juin 2023

Michael de Raadt, « Progresse Bar », [https://moodle.org/plugins/block\\_progress](https://moodle.org/plugins/block_progress), consulté le 14 juin 2023

« Moodle statistics », <https://moodle.net/stats/>, consulté le 14 juin 2023

« Maximum d'essais avec la même version de la série d'exercices, Paramétrer l'exercice », [https://wims.univ-cotedazur.fr/wims/wims.cgi?](https://wims.univ-cotedazur.fr/wims/wims.cgi?session=DN00FDA16F.2&+lang=en&+module=help%2Fteacher%2Fdocbeginner.fr&+cmd=reply&+job=read&+foldin=30..4#primi.4)

[session=DN00FDA16F.2&+lang=en&+module=help%2Fteacher](https://wims.univ-cotedazur.fr/wims/wims.cgi?session=DN00FDA16F.2&+lang=en&+module=help%2Fteacher%2Fdocbeginner.fr&+cmd=reply&+job=read&+foldin=30..4#primi.4)

[%2Fdocbeginner.fr&+cmd=reply&+job=read&+foldin=30..4#primi.4](https://wims.univ-cotedazur.fr/wims/wims.cgi?session=DN00FDA16F.2&+lang=en&+module=help%2Fteacher%2Fdocbeginner.fr&+cmd=reply&+job=read&+foldin=30..4#primi.4), consulté le 15 juin 2023

« Nombre d'exercices dans la série, Paramétrer l'exercice », <https://wims.univ-cotedazur.fr/wims/wims.cgi?session=DN00FDA16F.2&+lang=en&+module=help%2Fteacher%2Fdocbeginner.fr&+cmd=reply&+job=read&+foldin=30..4#primi.4>, consulté le 15 juin 2023

« NLP Progresse », [http://nlpprogress.com/english/dependency\\_parsing.html](http://nlpprogress.com/english/dependency_parsing.html), consulté le 6 juillet 2023.

« Quelques chiffres sur la déficience visuelle ». En ligne :

<https://aveuglesdefrance.org/quelques-chiffres-sur-la-deficience-visuelle/>, consulté le 10, août, 2023.

Stanza <https://stanfordnlp.github.io/stanza/>, consulté le 7 juillet 2023.

Spacy <https://spacy.io/>, consulté le 7 juillet 2023.

UDPipe , <https://lindat.mff.cuni.cz/services/udpipe/> consulté le 6 juillet 2023.

« *Université inclusive : le point sur les avancées en 2022* », 2022, 4 février. En ligne : <https://informations.handicap.fr/a-universite-inclusive-point-sur-avancees-2022-32303.php>, consulté le 10, août, 2023.

« *Universal Dependencies* », <https://universaldependencies.org/>, consulté le 5 juillet 2023.

« *Web Content Accessibility Guidelines (WCAG) 2.1* », <https://www.w3.org/TR/WCAG21/>, consulté le 12 juillet 2023.

« WIMS », <https://fr.wikipedia.org/wiki/WIMS>, consulté le 15 juin 2023

Xiao Gang, <https://sites.google.com/view/gangwims/xiao-et-wims> , consulté le 15 juin 2023.

Python for NLP <https://realpython.com/nltk-nlp-python/>, consulté le 19 juillet 2023.