Apprentissage d'inférences par édition d'arbres pour répondre à des questions

Brigitte Grau, Martin Gleize

Apprentissage d'inférences par édition d'arbres pour répondre à des questions

Martin Gleize* — **Brigitte Grau****

* *LIMSI-CNRS, Université Paris-Sud*
** *LIMSI-CNRS, ENSIIE*

RÉSUMÉ. *La sélection de réponse en recherche d'information précise met nécessairement en oeuvre un appariement de passages avec la question. Nous proposons un algorithme qui consiste à engendrer et apprendre les inférences utiles pour rapprocher les passages de texte à des couples (questions, réponse candidate). Ceux-ci sont sélectionnés au moyen d'une expansion lexicale utilisant WordNet et des vecteurs de mots. Ils sont représentés par leur arbre de dépendances syntaxique, enrichi au moyen de plusieurs ressources lexico-sémantiques dont WordNet et ConceptNet. Un algorithme de recherche en faisceau calcule des séquences d'édition transformant les passages en choix de réponse. Des traits calculés à partir de ces séquences d'édition sont utilisés pour construire deux classifieurs chargés respectivement de valider ou invalider chaque choix de réponse. Cette méthode a été évaluée sur la tâche "Entrance Exams" à CLEF 2015 qui propose de répondre à des QCMs sur des textes et a obtenu le deuxième meilleur score de précision des systèmes participants.*

ABSTRACT. *In order to answer question, we propose a matching algorithm that consists in generating and learning inferences needed to rely text passages to pairs (question, candidate answer). We first retrieve relevant passages, through lexical expansion involving WordNet and word vectors, that are enriches by lexico-semantic resources. Then a tree edit model is used on graph representations of the passages and answer choices to extract edit sequences. Finally, features are computed from those edit sequences and used in two machine-learned models, one for validating answers and one for invalidating answers, in order to take the final decision. This method was evaluated in the Entrance Exams task at QALD@CLEF, that consists in answering QCMs on given texts, yielding the second best precision score on the task.*

MOTS-CLÉS : *système de question-réponse, questions à choix multiples, algorithme d'édition d'arbres, inférence textuelle.*

KEYWORDS: *Question-answering system, Multiple Choice Questions, Tree edit algorithm, textual entailment.*

1. Introduction

Différentes solutions peuvent être mises en oeuvre pour rechercher la réponse à une question. On peut rechercher dans des textes ou bien dans des bases de questions dont on connaît déjà la réponse, dans des FAQ ou sur des sites communautaires. Mais quelle que soit la tâche, on en revient toujours à devoir apparier l'information exprimée par la question avec celle qui est contenue dans un passage de texte. Cet appariement peut s'exprimer comme un problème d'implication textuelle afin de valider, ou sélectionner, la réponse pertinente : le passage sélectionné implique-t-il l'hypothèse formée par la question ?

La sélection de réponse a donné lieu à des méthodes qui s'appuient sur différents niveaux d'analyse de la langue, et produisent de ce fait des représentations différentes des énoncés (questions et passages). Les premières ont défini des mesures de similarité globale sur une représentation de type "sac de mots" (Ferret *et al.*, 2001 ; Magnini *et al.*, 2002 ; Ittycheriah *et al.*, 2001) ou ont formulé la sélection de réponse comme un problème de classification ou de re-ordonnancement (Cui *et al.*, 2005 ; Grappy *et al.*, 2011) appris automatiquement. Afin de tenir compte des relations entre mots des énoncés lors de l'appariement, des modèles d'alignement d'arbres de dépendances (Wang *et al.*, 2007 ; Yao *et al.*, 2013) ou de leur transformation par éditions successives (Kouylekov *et al.*, 2007 ; Heilman et Smith, 2010 ; Moschitti *et al.*, 2007) ont été proposés. Cependant, les connaissances sémantiques, même si elles sont souvent prises en compte, sont ajoutées en dehors de ces modèles d'appariement.

Une véritable intégration de syntaxe et sémantique existe dans quelques travaux, comme le modèle de Wang et Manning (2010), qui définit des opérations d'édition avec inclusion de variations sémantiques sur les mots. Pour aller au delà du niveau lexical et élargir les types d'inférences envisagées, nous proposons d'enrichir les arbres de dépendances par des relations sémantiques entre couples de concepts. Ces relations nous sont fournies par la base de triplets ConceptNet, une ressource de connaissance de sens commun structurée. Un algorithme de recherche en faisceau calcule des séquences d'édition transformant les arbres des passages en ceux des choix de réponse. Des traits calculés à partir de ces séquences d'édition sont utilisés pour construire deux classifieurs chargés respectivement de valider ou invalider chaque choix de réponse.

Afin de rendre l'évaluation plus directe et pertinente, il est possible de fixer ces choix de réponses, en terme de leur nombre et de leur difficulté, dans le cadre de la réponse à des QCM de compréhension de texte. La conférence CLEF 2015 propose la tâche "Entrance Exam". Elle évalue les systèmes sur les mêmes QCM que les étudiants passent à l'entrée à l'université, avec l'hypothèse que différents types d'inférences soient nécessaires pour y répondre. Notre méthode a été évaluée sur cette tâche à CLEF 2015 et a obtenu le deuxième meilleur score avec une précision de 0,36. Ces travaux ont été publiés précédemment en anglais à la conférence sus-citée (Gleize et Grau, 2015).

2. État de l'art

Les travaux récents en question-réponse se focalisent sur la sélection d'un passage réponse, étant donné un ensemble de passages candidats. Les approches sacs de mots ont été améliorées sur cette tâche par des modèles qui cherchent à apprendre l'alignement latent entre les deux représentations, celle du passage et celle de la question. Différentes mesures ont été conçues reposant sur un décompte des alignements entre les relations de dépendance des deux fragments de texte. Wang *et al.* (2007) apprennent cet alignement par un modèle probabiliste génératif fondé sur une grammaire quasi-synchrone. Ce formalisme permet de tenir compte de transformations locales, entre des équivalences de syntagmes tels que "leader of France" et "French president". Ces modèles réalisent un alignement mot à mot où un mot de la question est associé à un mot du passage, ce qui ne permet pas de trouver des paraphrases sous-phrastiques de tailles différentes, ce qui a conduit Yao *et al.* (2013) à étendre un CRF par un modèle semi-markovien qui permet d'introduire cette flexibilité.

L'édition d'arbres de dépendances pour une tâche de sélection de passage a été proposée pour la première fois par (Punyakanok *et al.*, 2004). Elle a été intégrée à un système de validation de réponse par (Kouylekov *et al.*, 2007). Heilman et Smith (2010) en ont proposé une version évaluée sur la sélection de phrases réponses, qui apprend un classifieur sur des traits extraits des séquences d'opérations d'édition. Celles-ci sont produites par un algorithme glouton appliquant une fonction noyau sur les arbres pour évaluer leur similarité. Dans (Moschitti *et al.*, 2007), les auteurs proposent une fonction noyau sur les arbres qui étend la définition de sous-arbres similaires. Enfin, le modèle de Wang et Manning (2010) apprend l'alignement latent de deux arbres de dépendances en définissant des opérations d'édition qui incluent des transformations sémantiques, intégrant ainsi syntaxe et sémantique.

Ces modèles sont évalués[1] sur le jeu de test de (Wang *et al.*, 2007) qui contient 89 questions de TREC et où les réponses sont énoncées par une seule phrase.

De ce fait les modèles proposés cherchent à aligner deux représentations de phrases, alors que souvent les informations données dans la question sont réparties dans des passages de plusieurs phrases. Dans (Sachan *et al.*, 2015), les auteurs proposent un modèle d'alignement intégrant, entre autres, cette contrainte. Il a été conçu pour la tâche de compréhension de texte, "Machine Reading", et est évalué sur le corpus MCTest[2] qui est un ensemble de QCMs construit par crowdsourcing.

1. Voir `http://aclweb.org/aclwiki/index.php?title=Question_Answering_State_of_the_art` pour obtenir les scores des différents systèmes
2. http ://research.microsoft.com/mct

3. Présentation générale de la méthode

Le but général de notre système est de choisir la réponse correcte parmi celles qui sont proposées. Voici un extrait de texte, suivi par un item du QCM, composé d'une question et de proposition de réponses dont une seule est correcte.

Texte : It was early morning. Peter Corbett helped Mark Wellman out of his wheelchair and onto the ground. They stood before El Capitan, a huge mass of rock almost three-quarters of a mile high in California's beautiful Yosemite Valley. It had been Mark's dream to climb El Capitan for as long as he could remember. But how could a person without the use of his legs hope to try to climb the highest vertical cliff on earth ? [...]

Question : What had Mark Wellman long desired to do ?

1) To accomplish one of the most difficult rock climbs in the world.(correcte)

2) To be the first to conquer El Capitan.

3) To climb the highest mountain in California.

4) To help his friend Peter climb El Capitan.

La réponse correcte provient généralement d'une formulation assez éloignée de ce qui la justifie dans le texte et repose sur la capacité à trouver les inférences nécessaires, inférences qui relèvent de connaissances générales et non de connaissances encyclopédiques ou de connaissances d'un domaine. Par ailleurs, les réponses incorrectes peuvent être partiellement retrouvées dans le texte, et il nous a paru intéressant d'être en mesure d'invalider les réponses fausses. Généralement les systèmes cherchent à valider des réponses, i.e. trouver les caractéristiques permettant de rapprocher deux extraits de textes, et non à invalider des réponses, i.e. trouver des caractéristiques qui rendent compte d'un changement de sens entre deux passages.

Notre objectif est de valider des réponses correctes sans les invalider, et inversement d'invalider des réponses fausses, sans les valider, en référence à un passage support. La sélection de ces passages est effectué en rapport avec la question. En effet, en étudiant les textes, il nous est paru que la question amène à délimiter un espace de recherche dans lequel les candidats réponse, corrects ou non, peuvent être analysés.

Le système que nous proposons est donc composé de quatre modules (cf. figure 1) : recherche de passages, enrichissement sémantique des graphes de dépendances, édition de graphes par une recherche en faisceau, sélection de réponse reposant sur l'apprentissage de classifieurs pour valider/invalider des réponses.

4. Sélection de passages

La sélection de passage répond à deux objectifs conjoints : réduire l'espace de recherche de manière à pouvoir appliquer des processus coûteux en temps, et sélectionner un contexte en relation avec le couple (question, réponse candidate) pour justifier

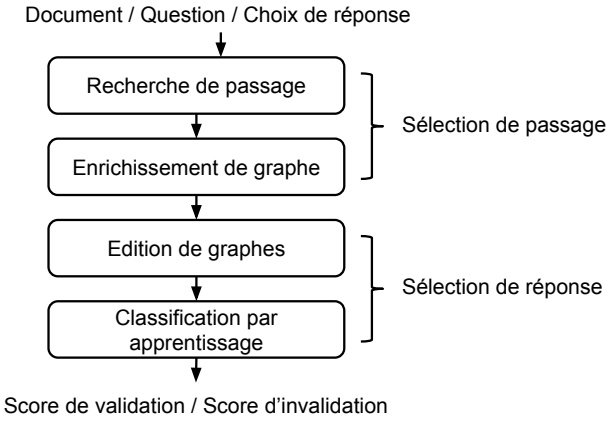

Figure 1. *Architecture du système*

ou non une réponse. L'objectif est de sélectionner des passages qui rendent compte au mieux du sens du couple. C'est pourquoi nous favorisons les passages qui permettent un alignement avec le maximum de mots du couple requête.

4.1. *Recherche de passages*

Les mots du couple (question, réponse candidate) forment la requête de base, sous forme de lemmes. Comme il est rare qu'ils soient présents tel quels dans le texte, nous appliquons une forme d'expansion. Nous enrichissons chaque lemme avec les informations de coréférence et les lemmes en relation sémantique avec lui dans WordNet (synonymes, antonymes, hyperonymes, hyponymes). Si un mot ne peut toujours pas être relié au texte, nous calculons une mesure de similarité fondée sur les représentations distribuées construites par (Huang *et al.*, 2012) et nous associons un vecteur de 50 traits à chacun des lemmes des mots du texte et de la requête selon la méthode proposée par l'auteur. Nous apparions les mots de la requête avec les mots du passage ayant la plus forte similarité par la fonction $cosinus$. Nous tenons compte des bi-termes en les représentant par la somme de leurs vecteurs, ce qui nous permet des alignements 1-2, 2-1 et 2-2.

Les passages sont délimitées par une fenêtre glissante de taille fixe[3] et sont ordonnés en fonction de leur score, calculé par la formule 1. Ils sont ensuite étendus aux frontières de phrases.

3. La taille choisit permet de capturer 3 phrases environ, nombre qui résulte d'une étude du corpus

$$score(passage) = \frac{\#matchedWords}{\#queryWords} \times \sum_{i=1}^{n-1} \frac{score(w_i) + score(w_{i+1})}{dist(i, i+1)^2} \quad [1]$$

Chaque mot du passage $w_i \in \{w_1, ..., w_n\}$ apparié avec un mot de la requête reçoit un score d'alignement $score(w_i)$ (1 si les lemmes sont égaux, 0,9 si ils sont synonymes dans WordNet, 0,8 pour les autres relations de WordNet et sinon le score de similarité de leur vecteurs, pondéré par leur ISF (Inverse Sentence Frequency). La formule est divisée par le carré de la distance des mots trouvés dans le passage afin de favoriser les passages les plus compacts. Pour tenir compte de l'absence potentielle de mots de la requête, nous multiplions le score du passage par la proportion de mots de la requête qui lui sont reliés.

Cette méthode retrouve un grand nombre de courts passages, souvent en position de recouvrement, mais l'algorithme en faisceau que nous appliquons ensuite pour les analyser permet de traiter de nombreux passages.

4.2. *Représentation et enrichissement des passages*

Les passages sont représentés par la structure syntaxique des phrases qui les composent, i.e. l'arbre de dépendances, reliées suivant leur séquence dans le passage. Ils sont ensuite enrichis par des connaissances permettant de réaliser des inférences. Le but de cet enrichissement est de pouvoir appliquer les règles d'édition sur les différents types de connaissances et opérer ainsi des inférences syntaxiques ou sémantiques en se reposant sur la même méthode.

Avant de décrire l'enrichissement des graphes, nous allons préciser les prétraitements appliqués au texte.

4.2.1. *Prétraitement des textes*

Nous utilisons la suite CoreNLP de Stanford pour analyser les textes. Chaque phrase du texte, les questions et les réponses sont annotées par les catégories morpho-syntaxique des mots (Toutanova *et al.*, 2003). Les graphes de dépendance sont engendrés par l'application de l'analyseur de (Klein et Manning, 2003). Un outil de résolution de co-références (Recasens *et al.*, 2013) est appliqué sur le texte et sur les item des QCMs. En effet, les textes que nous traitons sont souvent de courts récits, faisant usage de nombreux pronoms personnels. De plus, tous les types de textes font usage d'anaphores nominales. On trouve souvent des références à un personnage sous différentes appellations générales. Une particularité des QCM consiste à employer les termes "the author" ou "the writer" dans les items (questions et réponses) quand le texte est rédigé à la première personne. Nous ajoutons de ce fait un traitement spécifique de ces tournures afin d'unifier les deux vocables. Notons que nous n'avons pas appliqué de module de reconnaissance d'entité nommées, les textes en comportant peu, et, en général, les questions posées n'attendent pas une réponse de ce type.

4.2.2. *Enrichissement des graphes*

Afin de construire un graphe pour le passage, les racines des graphes de dépendances des phrases sont reliés avec un arc *followed-by* qui matérialise le fait qu'une phrase en suit une autre, et maintient la cohérence du passage.

L'enrichissement est fondé sur la ressource ConceptNet (Liu et Singh, 2004). ConceptNet est une base de triplets sémantiques construite manuellement qui comporte des connaissances générales de sens commun, destinée à être utilisée pour la compréhension automatique de textes. Les noeuds d'un triplet sont des mots ou des expressions courtes de la langue, et la relation entre eux est étiquetée. Les noeuds sont dénommés "concept" mais il est plus juste de les considérer comme des termes. Par exemple, ConceptNet contient des connaissances de base assez communes comme *MotivatedByGoal(learn, knowledge)* : on apprend dans le but d'acquérir des connaissances, ou *UsedFor(saxophone, jazz)* : un saxophone est utilisé pour le jazz. D'autres bases de triplet plus connues comme DBpedia ou YAGO se concentrent sur les entités nommées, qui ne sont ici qu'un problème rare.

Notre hypothèse est que ce type de relation d'inférence entre termes correspond aux connaissances mobilisées pour comprendre les textes présents dans les QCM. Ces connaissances de sens commun sont généralement maîtrisées par les êtres humains, mais sont difficiles à saisir par une machine. C'est pourquoi nous avons voulu enrichir les graphes des passages par ces relations, de manière à pouvoir combler les pas de raisonnement manquant pour passer d'un mot à un autre, ou d'un sous-graphe à un autre.

Les *Concepts* de ConceptNet sont principalement nommés par des mots uniques, comme "saxophone" ou "jazz". Il est donc aisé de relier ses triplets à nos graphes, ce lien se faisant sur les noeuds communs. Il n'est cependant pas aussi simple d'intégrer les relations, car leur étiquette est généralement formée de plusieurs mots, comme "UsedFor", "MotivatedByGoal", et nous voulons obtenir une représentation unifiée des passages. C'est pourquoi nous avons rattaché aux graphes des phrases les arbres d'analyse des formes de surface associées à la relation. Ces formes de surface sont les phrases originales en langue naturelle qui expriment la relation, comme "a saxophone is used for jazz". Nous rattachons l'arbre d'analyse de ces expressions chaque fois qu'un concept possède un mot de tête présent dans le graphe de dépendance du passage. Nous avons retenu de ConceptNet les relations exprimant une implication textuelle, soit : IsA, PartOf, MemberOf, UsedFor, CapableOf, Causes, HasPrerequisite, MotivatedByGoal, Desires.

5. Sélection de réponse

Notre objectif est de caractériser une séquence de transformations appliquées au passage permettant d'obtenir une réponse candidate. Ces transformations portent sur le graphe enrichi du passage, et les transformations en sont des éditions ce qui nous a conduit à proposer un algorithme d'édition d'arbre. Les opérations d'édition appli-

quées permettront ensuite de fournir des traits à deux classifieurs, l'un pour valider des réponses, l'autre pour invalider des réponses.

5.1. *Edition de graphes par une recherche en faisceau*

5.1.1. *Opérations d'édition*

Le principe de l'algorithme consiste à appliquer au graphe différentes opérations d'édition itérativement, le modifiant ainsi à chaque itération, de manière à ce que le graphe soit de plus en plus proche de l'arbre cible représentant l'hypothèse à atteindre, i.e. une réponse candidate. La liste d'éditions consécutives constitue une *séquence d'édition*. La table 1 présente les opérations d'édition choisies. Les opérations "delete", "insert", "rename" sont les opérations minimales d'un algorithme d'édition. L'opération "move" est plus inhabituelle car elle permet d'effectuer d'importants changements dans la phrase juste en rattachant un noeud à un nouveau noeud père. Si ce noeud est la racine d'un sous-arbre ayant des descendants, cela fait déplacer un large extrait de texte. Les opérations n'utilisent pas les étiquettes des relations de dépendance, mais celles-ci seront utilisées dans les traits extraits.

La figure 2 présente un exemple de l'application consécutive de trois opérations.

Opération d'édition	Description
Delete(d : Tree)	Supprime le noeud *d* et le remplace par son fils
Insert(i : Word, p : Tree)	Insère le mot *i* sous son nouveau père *p*.
Rename(t : Tree, w : Word)	Remplace le mot attaché au noeud *t* par *w*.
Move(m : Tree, op : Tree, np : Tree)	Déplace le sous-arbre *m* sous *op* sous *np*.

Tableau 1. *Opérations d'édition d'arbre de dépendance*

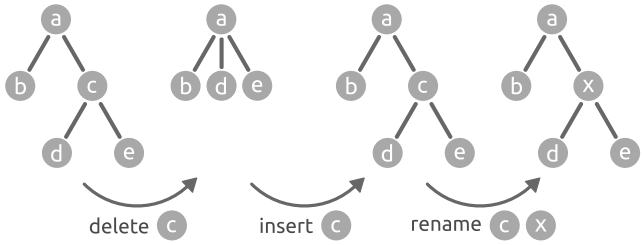

Figure 2. *Exemple d'application successive d'opérations d'édition*

L'exploration des éditions à appliquer se ramène à un problème de recherche de chemin dans un graphe. Cependant il n'est pas possible d'appliquer toutes les transformations possibles à chaque itération, pour des problèmes de complexité en temps et en espace. Aussi il est nécessaire de savoir sélectionner les plus prometteuses.

5.1.2. *Recherche en faisceau*

Le problème principal de l'algorithme calculant les éditions est de savoir quand appliquer quelle opération d'édition, savoir quel élément ajouter ? Nous appliquons des heuristiques simples pour limiter le nombre d'opérations à considérer (par exemple ne pas insérer de noeud qui n'aide pas à atteindre la cible, ne pas renommer ou supprimer un noeud qui est présent dans l'arbre cible). Cependant un grand nombre d'opérations restent possibles. Aussi, au lieu de choisir à chaque pas les opérations à appliquer, nous les appliquons toutes et choisissons ensuite de développer les plus prometteuses en nous appuyant sur une bonne heuristique.

La *recherche en faisceau* opère une recherche en largeur d'abord, ce qui minimise l'espace mémoire requis. A chaque pas, toutes les éditions sont appliquées et les arbres obtenus sont ordonnés selon un coût heuristique. Cependant seul un nombre limité d'arbres est conservé à chaque étape, qui correspond à la largeur du faisceau. Cette méthode permet d'ajuster, *via* ce paramètre, la probabilité de trouver des séquences d'édition utiles et les coûts en temps et en espace.

L'heuristique doit permettre d'estimer la proximité du noeud exploré avec la cible. Nous avons implémenté à cette fin la fonction noyau sur les arbres partiels définie dans (Moschitti, 2006) pour calculer la similarité entre l'arbre courant et l'arbre cible. Comme toute fonction noyau d'arbre, elle calcule le nombre de sous-arbres communs entre les deux arbres mais cette version s'applique à des arbres n-aires, ce que sont les arbres de dépendance. Le calcul du noyau est normalisé par $\tilde{K}(x,y) = \frac{K(x,y)}{\sqrt{K(x,x)}\sqrt{K(y,y)}}$, pour K le noyau et x, y les arbres.

5.1.3. *Algorithme*

L'algorithme développe un graphe représentant une partie de l'espace de recherche, où chaque noeud est un arbre et les arcs sont les opérations d'édition appliquées. Au début de l'algorithme, ce graphe est initialisé par les p arbres des passages sélectionnés qui ont été enrichis. L'arbre cible est celui de la réponse lorsqu'elle est sous forme de phrase ou celui de la question plus la réponse lorsque la réponse est la fin d'une phrase question. Ensuite chaque opération d'édition est appliquée à chaque passage :

– "Insert" et "Rename" ne peuvent ajouter que des noeuds qui existent dans l'arbre cible ;

– Move peut uniquement déplacer un noeud sous un noeud père tel que le lien parent $\rightarrow$ enfant est présent dans l'arbre cible.

Les arbres modifiés sont ajoutés au graphe, et l'heuristique est calculée pour chacun d'eux, permettant de les ordonner. Les n meilleurs sont gardés et l'algorithme est réitéré. Il s'arrête quand k séquences d'édition différentes ont été trouvées ou après l itérations. Dans nos expérimentations, nous avons retenu : $p = 10$, $n = 50$, $k = 10$ et $l = 200$.

5.2. *Validation et invalidation de réponses*

L'objectif est de classer les séquences d'édition par l'apprentissage de deux classifieurs : l'un pour décider si la séquence permet de valider une réponse correcte, l'autre pour décider si elle permet d'invalider une réponse fausse. Il s'agit donc d'en extraire des traits caractéristiques qui permettent ces classifications. En pratique, les traits et les algorithmes sont les mêmes pour apprendre les deux classifieurs. Seuls les corpus d'apprentissage sont différents.

5.2.1. *Extraction des traits*

La plupart des traits dénombrent des opérations d'éditions particulières dans la séquence, sur des unigrammes ou des bigrammes, et sont résumés table 2. Les informations présentes dans les arbres de dépendances du passage et non utilisées lors de la recherche en faisceau sont utilisées à cette étape : les relations de dépendances, les co-références ou le fait qu'une édition s'applique sur un concept de ConceptNet ou de WordNet.

5.2.2. *Apprentissage des classifieurs*

Les deux modèles, pour la validation et l'invalidation de réponse candidate[4], sont appris à partir des mêmes traits mais sur des corpus d'entraînement différents. Nous voulons éviter d'apprendre comment transformer un passage de texte pris au hasard en une réponse prise au hasard aussi. En tant que lecteur, on ne peut valider une réponse en lisant un quelconque passage du texte, ni en rejeter une si le passage support n'est en rien relié à la question et à la réponse. Aussi, nous avons annoté manuellement la pertinence de passages relativement à un couple (question, réponse) afin de construire les corpus d'apprentissage. Un *passage pertinent* est défini comme l'extrait minimum qui permet d'exprimer à la fois la question et les éléments de réponse, i.e. de justifier la réponse si elle est correcte ou d'invalider celle-ci. Les réponses incorrectes ne sont bien sûr généralement pas exprimées dans leur totalité. Parfois même, une réponse n'est pas exprimée du tout dans le texte. Deux réponses liées à la même question peuvent partager le même passage justificatif.

Nous construisons les corpus d'apprentissage en suivant la sémantique décrite figure 3. Un passage est associé à un choix de réponse ; aussi passage $choixX$ fait référence au passage associé à n'importe quel choix de réponse incorrecte (X) et passage $choix1$ à une réponse incorrecte particulière. Pour la validation de réponse, les exemples positifs sont les couples (passage, réponse correcte). Pour l'invalidation de réponse, les exemple positifs sont les couples (passage correct, réponse incorrecte) et (passage pertinent pour la réponse incorrecte considérée, réponse considérée), et ce pour chacune des réponses incorrectes. Les autres combinaisons forment les exemples négatifs pour les deux classifieurs. Les couples positifs pour un classifieur donnent lieu à des couples négatifs pour l'autre. Ces combinaisons, surtout pour les réponses

4. Nous appelons réponse les choix de réponses, i.e. les réponses candidates.

Trait	Description
editTotal	Nombre total d'éditions dans la séquence
deleteTotal deleteVerb deleteNoun deleteProperNoun deleteSubject deleteObject deleteRoot deleteNegation deleteConceptNet	Nombre total d'opérations de suppression, édition qui supprime un verbe, un nom, un nom propre, un sujet (indiqué par la relation de dépendance subj), un complément d'objet, la racine d'un arbre, une négation (indiquée par la relation de dépendance neg) et un élément ajouté au graphe à partir de ConceptNet
insertTotal insertVerb insertNoun insertProperNoun insertNegation	Analogue aux informations ci-dessus, pour les opérations d'insertion
renameTotal renameVerb ... renameSyn renameAnt renameHypHyp renameStrongWordVectorSim renameCoref renameNonCoref	Analogue aux informations ci-dessus, pour les opérations de renommage + les éditions qui renomme un mot en son synonyme, ou son antonyme ou son hyperonyme/hyponyme dans WordNet, édition qui remplace un mot par un autre trouvé fortement similaire selon leurs vecteurs (au dessus d'un seuil fixé empiriquement), éditions qui renomment un pronom en son référent d'après la résolution des coréférence de Stanford, et éditions qui renomment un pronom par d'autres référents
moveTotal moveVerb ... moveConceptNet moveMoreThan2Nodes	Analogue aux informations ci-dessus, pour les opérations de déplacement + les éditions qui déplacent plus de 2 noeuds
Combinaisons sur les bigrammes	Nombre de paires d'éditions successives dans la séquence
dependencyEditSequence	Nombre de paires d'éditions successives appliquées à 2 noeuds d'une relation de dépendance
originalTotal originalVerb ...	Proportion de mots, verbes, noms, noms propres originaux qui **ne sont pas** édités dans la séquence

Tableau 2. *Traits extraits d'une séquence d'édition*

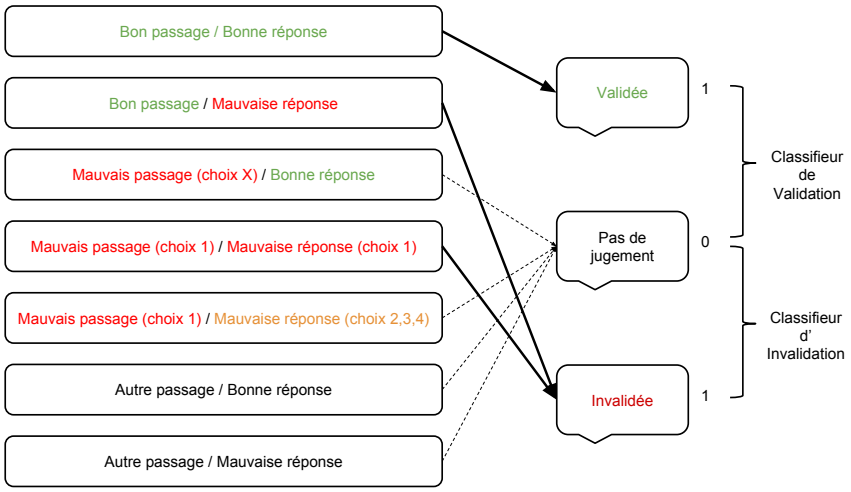

Figure 3. *Semantique des paires (passage, réponse)*

incorrectes, signifient que le passage choisi est suffisant pour faire état d'une incohérence de la réponse par rapport à la question.

6. Expérimentations et résultats

6.1. *Tâche "Entrance Exams" à QALD@CLEF 2015 : données et évaluation*

Nos données sont celles distribuées pour la tâche "Entrance Exams" à CLEF 2015. Elles sont extraites des examens d'entrée à l'université de Tokyo, pour tester la maîtrise de l'anglais langue étrangère. Le jeu d'entraînement est composé des jeux de test 2013 et 2014. Chacun comporte 12 textes, et 4 à 6 items par QCM, soit environ 120 questions auxquelles il faut répondre. Le test de 2015 comporte 19 textes et un total de 89 questions. Il y a 4 choix de réponse par item.

Les systèmes sont évalués en terme de précision : la fraction de réponses correctes.

6.2. *Sélection d'une réponse*

Les algorithmes renvoient des scores pour les séquences d'édition retenues par couple (question, réponse), soit un score de validation et un score d'invalidation, et nous retenons la séquence ayant la plus forte valeur parmi les *max*(`validationScore`, `invalidationScore`). Idéalement nous voudrions obtenir une séquence d'édition caractéristique d'un processus de validation ou invalidation fiable de manière à pouvoir classer la réponse comme correcte ou incor-

recte. Ensuite la réponse dont la séquence d'édition a la plus grande valeur de `validationScore` − `invalidationScore` est choisie : nous choisissons comme réponse celle pour laquelle la séparation entre validation et invalidation est la plus nette.

6.3. *Résultats*

Nous avons expérimenté deux modèles de classification, la régression logistique et les forêts d'arbre aléatoires, implémentées dans Weka (Hall *et al.*, 2009). La table 3 montre les résultats obtenus pour chaque modèle, et les résultats obtenus par les autres participants à l'évaluation. Une précision de 0,36 nous place en 2ème position avec le modèle des forêts aléatoires. La baseline est 0,25 et correspond à répondre au hasard. Le premier système repose sur l'utilisation de processus et de connaissances propriétaires (Laurent *et al.*, 2015).

	# questions correctes	# questions erronées	# tests avec précision ≥ 0.5	précision
Forêt aléatoire	32	57	8	0,36
Régression logistique	28	61	4	0,31
Synapse	52	37	16	0,58
cicnlp	27	62	6	0,30
NTUNLG	26	63	6	0,29
CoMiC	26	63	5	0,29

Tableau 3. *Résultats*

6.4. *Analyse d'erreurs*

6.4.1. *Analyse quantitative*

La tendance générale de notre système est de se comporter mieux quand les séquences d'édition sont courtes, avec une précision de 40% quand les séquences choisies sont de longueur < 6 en moyenne.

Nous avons aussi effectué des tests concernant les ressources utilisées pour enrichir les graphes, WordNet et ConcepNet. Nous avons essayé d'enrichir aussi les réponses, dans le but de permettre des inférences d'ordre 2 (un mot est lié à un autre *via* un mot intermédiaire). Mais cela a dégradé les performances. Nous avons aussi effectué des tests d'ablation, d'abord ConceptNet puis WordNet. Enlever l'une ou l'autre ressource montre une légère baisse, mais pas autant que l'on pourrait l'espérer. Des études plus poussées sont nécessaires pour conclure.

6.4.2. *Analyse qualitative*

Il est difficile d'effectuer une réelle analyse qualitative aussi nous allons juste donner quelques exemples (passage, réponse fournie) illustratifs de la tâche.

Dans le passage suivant, le système est trompé par la réponse 3, proche du texte, et ConceptNet ne permet pas de relier "held" à "trapped", et "its original nature" au passage (le lien figure plus loin dans le texte).

Several years ago, certain scientists developed a way of investigating the nature of the atmosphere of the past by studying air caught in the ice around the North or South Pole. According to their theory, when snow falls, air is trapped between the snowflakes. The snow turns to ice with the air still inside.
Certain scientists claimed that 1) atmospheric gases increase the yearly amount of snow 2) falling snowflakes change the chemical balance of the air 3) the action of atmospheric gases causes snow to turn into ice **4)** the air held between snowflakes keeps its original nature (correct)

Dans l'exemple suivant notre système choisit la réponse 3 au lieu de 1 qui aurait pu l'être si "wrong" avait pu être lié à "mistake". Mais dans ConceptNet cette relation est du type *RelatedTo*, que nous n'avons pas retenue car trop imprécise.

Everyone stared. That was embarrassing enough, but it was worse when I finished my coffee and got ready to leave. My face went red - as red as his hair - when I realized I'd made a mistake.
The woman's face turned red **1)** because she realized that she had been quite wrong about the boy (correct) 2) because she realized that the boy was poor and hungry 3) because she saw everyone staring at her 4) because she hated being shouted at

Dans ces deux cas, une meilleure caractérisation du passage correct aurait été utile, car dans le premier nous omettons la phrase qui contient la réponse correcte et dans le second le passage sélectionné apparaît bien avant le passage lié à la question+réponse correcte. Mais nous n'avons pas voulu intégrer le fait que généralement l'ordre des questions des QCM suit l'ordre des phrases, dans la mesure où nous sommes intéressés par décider de l'adéquation d'un passage sélectionné pour valider ou non une réponse.

Enfin le dernier exemple montre une réponse correcte à laquelle le système répond majoritairement par invalidation. Les réponses 1 et 4 sont nettement invalidées, grâce respectivement à la présence de la négation et de la 1ère phrase exprimant le contraire. La réponse 2 est moins nette, mais avec une tendance pour l'invalidation. La réponse 3 est finalement choisie par défaut, car elle n'est ni validée ni invalidée.

Kate was an energetic woman who expected people always to be doing something, and she found plenty of jobs for Fred to do. This made him feel part of the household, but now he really wanted to be able to sit and reflect on the events of his life. If he had continued to live alone, he would have had the time to do this to his heart's content. One afternoon he felt he simply had to get away from the house. "I'm going for a walk," he said, closing the door behind him. Leaving the town, he walked across the fields and followed a slow-moving stream toward the hills. After a while he came to a pool in the stream under some trees. Here, he thought, was a place he could come to when he needed to reflect on the past. Although the stream seemed unlikely to have any fish, he would simply tell Kate he had found a place to go fishing. When he mentioned the stream that night, his son-in-law, Jim, said in disbelief, "There aren't any fish there. That stream runs dry half the summer."
Why did Fred tell Kate that he had found a place to go fishing ? 1) He didn't feel part of the household with Kate and Jim. 2) He enjoyed fishing very much and was glad to be able to do it again. **3)** He wanted a way to leave the house without hurting Kate's feelings. 4) He was bored in the house because there were few things to do.

7. Conclusion

Nous avons proposé un algorithme permettant d'apparier deux textes afin de décider si l'un implique l'autre. Dans le but de tenir compte de variations syntaxiques et sémantiques, nous avons représenté le texte à apparier par un graphe d'arbres de dépendances, enrichi par des informations sémantiques issues de WordNet, de représentations distribuées et d'arbres issus de ConceptNet pour tenter de réaliser des inférences autres que de la similarité sémantique de mots.

Nous avons intégré cet algorithme dans un système destiné à valider et invalider des réponses afin de répondre à des questions de QCM dans le cadre de l'évaluation CLEF "Entrance Exams". Nous avons obtenu la 2ème place à cette évaluation en 2015.

Dans le futur nous voulons améliorer l'enrichissement de notre graphe, car cette méthode permet de traiter de manière unifiée les variations linguistiques de toute nature, mais la couverture actuelle des ressources n'est pas encore suffisante. Nous envisageons d'y ajouter des paraphrases. Nous envisageons aussi de développer un meta-classifieur pour prendre la décision finale par question.

8. Bibliographie

Cui H., Sun R., Li K., Kan M.-Y., Chua T.-S., « Question answering passage retrieval using dependency relations », *SIGIR*, 2005.

Ferret O., Grau B., Hurault-Plantet M., Illouz G., Jacquemin C., « Document selection refinement based on linguistic features for QALC, a question answering system », *RANLP*, 2001.

Gleize M., Grau B., « LIMSI-CNRS@CLEF 2015 : Tree Edit Beam Search for Multiple Choice Question Answering », *Working Notes of CLEF 2015 - Conference and Labs of the Evaluation forum, Toulouse, France, September 8-11, 2015.*, 2015.

Grappy A., Grau B., Falco M.-H., Ligozat A.-L., Robba I., Vilnat A., « Selecting answers to questions from Web documents by a robust validation process », *WI*, 2011.

Hall M., Frank E., Holmes G., Pfahringer B., Reutemann P., Witten I. H., « The WEKA data mining software : an update », *ACM SIGKDD explorations newsletter*, vol. 11, n° 1, p. 10-18, 2009.

Heilman M., Smith N. A., « Tree edit models for recognizing textual entailments, paraphrases, and answers to questions », *NAACL*, 2010.

Huang E. H., Socher R., Manning C. D., Ng A. Y., « Improving word representations via global context and multiple word prototypes », *ACL*, p. 873-882, 2012.

Ittycheriah A., Franz M., Roukos S., « IBM's Statistical Question Answering System-TREC-10. », *TREC*, 2001.

Klein D., Manning C. D., « Accurate unlexicalized parsing », *ACL*, p. 423-430, 2003.

Kouylekov M., Negri M., Magnini B., Coppola B., « Towards entailment-based question answering : ITC-irst at CLEF 2006 », *Evaluation of Multilingual and Multi-modal Information Retrieval*, Springer, p. 526-536, 2007.

Laurent D., Chardon B., Nègre S., Pradel C., Séguéla P., « Reading comprehension at Entrance Exams 2015 », *CLEF 2015 Working Notes*, 2015.

Liu H., Singh P., « ConceptNet—a practical commonsense reasoning tool-kit », *BT technology journal*, vol. 22, n° 4, p. 211-226, 2004.

Magnini B., Negri M., Prevete R., Tanev H., « Mining Knowledge from Repeated Co-Occurrences : DIOGENE at TREC 2002. », *TREC*, 2002.

Moschitti A., « Efficient convolution kernels for dependency and constituent syntactic trees », *Machine Learning : ECML 2006*, Springer, p. 318-329, 2006.

Moschitti A., Quarteroni S., Basili R., Manandhar S., « Exploiting syntactic and shallow semantic kernels for question answer classification », *ACL*, 2007.

Punyakanok V., Roth D., Yih W.-t., « Mapping dependencies trees : An application to question answering », *Proceedings of AI&Math 2004*, p. 1-10, 2004.

Recasens M., de Marneffe M.-C., Potts C., « The life and death of discourse entities : Identifying singleton mentions », *NAACL-HLT*, p. 627-633, 2013.

Sachan M., Dubey A., Xing E. P., Richardson M., « Learning Answer-Entailing Structures for Machine Comprehension », *ACL*, 2015.

Toutanova K., Klein D., Manning C. D., Singer Y., « Feature-rich part-of-speech tagging with a cyclic dependency network », *ACL-HLT*, Association for Computational Linguistics, p. 173-180, 2003.

Wang M., Manning C. D., « Probabilistic tree-edit models with structured latent variables for textual entailment and question answering », *COLING*, 2010.

Wang M., Smith N. A., Mitamura T., « What is the Jeopardy Model ? A Quasi-Synchronous Grammar for QA. », *EMNLP-CoNLL*, vol. 7, p. 22-32, 2007.

Yao X., Van Durme B., Callison-Burch C., Clark P., « Semi-Markov Phrase-Based Monolingual Alignment. », *EMNLP*, p. 590-600, 2013.