MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE SAAD DAHLEB – BLIDA 01

FACULTE DES SCIENCES

DEPARTEMENT D'INFORMATIQUE

MEMOIRE DE MASTER II

Spécialité : Ingénierie des Logiciels

THEME

LES RESEAUX DE NEURONES POUR LA GENERATION AUTOMATIQUE DE PARAPHRASES

Présenté par :

- ✓ **M. HAMEL Oussama**
- ✓ **Mlle LAMARI Selena**

Promotrice :

- ✓ **Mme OUAHRANI. L**

Devant le jury composé de :

- ✓ **M. BALA. M** — **Président**
- ✓ **M. FERFERA. S** — **Examinateur**

:

Soutenu le : 08/09/2020

Remerciements

Ce travail est l'aboutissement d'un dur labeur et de beaucoup de sacrifices ; nous remercions tout d'abord Dieu tout puissant de nous avoir donné le courage, la force et la patience d'achever ce travail.

Nous tenons à remercier également toutes les personnes qui ont contribué de près ou de loin à l'élaboration de ce travail, et à la réalisation de ce mémoire.

Nous remercions notre promotrice Madame L. Ouahrani, enseignante à l'Université Blida 1, elle nous a guidés dans notre travail et nous a aidés à trouver les solutions pour avancer. Nous la remercions également pour sa disponibilité et la qualité de ses conseils.

Nos remerciements vont aussi aux membres du jury qui ont pris de leur temps pour juger ce travail.

Nous souhaitons adresser nos remerciements les plus sincères au corps professoral et administratif de l'Université Blida 1, pour la richesse et la qualité de leur enseignement, et qui déploient de grands efforts pour assurer à leurs étudiants une formation actualisée.

Nos remerciements vont également à M. Maamar Missoum & M. Moustafa Messeded enseignants chercheurs au département d'anglais de l'université Blida 2, M. Amirat Omar traducteur officiel auprès de la cours de Bejaia, M. Taha Zerrouki enseignant chercheur à l'université de Bouira ainsi que Mme Zahra Fatma Zohra enseignant chercheur à l'université Blida 1 pour leur précieuse collaboration en effectuant une expertise humaine sur nos jeux de tests générés automatiquement aussi bien pour l'arabe que pour l'anglais.

Un grand merci à nos chers parents ainsi que les membres des familles Hamel et Lamari, pour leur amour, leurs conseils ainsi que leur soutien inconditionnel, à la fois moral et économique, qui nous a permis de réaliser les études que nous voulions et par conséquent ce mémoire.

Enfin, nous voudrions exprimer notre reconnaissance envers les amis et collègues qui nous ont apporté leur soutien moral et intellectuel tout au long de notre démarche.

RESUME

La génération de paraphrases est un problème important dans le Traitement Automatique de la Langue(TAL), spécialement dans les systèmes de questions/réponses comme les systèmes d'évaluation automatique des réponses courtes (ASAG pour Automatic Short Answer Grading).

Dans notre travail, nous abordons le problème de génération automatique de paraphrases pour la langue Arabe afin de les intégrer dans un système ASAG dédiée à cette langue. Nous avons proposé trois modèles basés sur le Deep Learning, le premier est un Modèle Bi-LSTM considéré comme modèle de base, le second est un encodeur/décodeur, le dernier modèle est un encodeur/décodeur avec mécanisme d'attention qu'on a nommé EDAM.

Nos tests ont été conduits dans les deux langues l'arabe et l'anglais en utilisant deux datasets, l'un en arabe, l'autre en anglais. Le deuxième a été choisi pour confirmer nos résultats obtenus pour l'Arabe. Une évaluation quantitative des approches proposées démontre l'efficacité du 3ème modèle nommé EDAM (Encoder-Decoder with an Attention Mechanism)

Une évaluation qualitative humaine sur un échantillon de paraphrases généré par notre approche, confirme que la qualité des paraphrases générées est très bonne sémantiquement et syntaxiquement.

Notre générateur EDAM a été intégré au sein de plusieurs variantes d'un système ASAG. **La précision** de ces systèmes s'est nettement améliorée. Cependant, le problème de manque de ressources en langue arabe constitue encore un défi à soulever pour l'évaluation automatique et pour toute autre activité du TAL Arabe en général.

Mots clés : Evaluation automatique des réponses courtes, Paraphrase, Apprentissage profond, Dataset.

ABSTRACT

The generation of paraphrases is a significant problem in NLP, especially in question/answers systems such as Automatic Short Answer Grading (ASAG) systems.

In our work, we address the problem of the automatic generation of paraphrases for the Arabic language in order to integrate it into an ASAG system. We proposed three models based on Deep Learning. The first is a Bi-LSTM model considered as a basic model. The second is an encoder / decoder. The last model is an encoder / decoder with an attention mechanism that we have named EDAM (Encoder-Decoder with an Attention Mechanism).

We conducted our tests in both Arabic and English languages using two datasets, one in Arabic, the other in English. The second was chosen as part of confirming our obtained results in Arabic. A quantitative evaluation of the proposed approaches demonstrates the effectiveness of the 3rd model called EDAM.

A human qualitative evaluation on a sample of paraphrases generated by our approach confirms that the quality of the generated paraphrases is very good semantically and syntactically.

Our EDAM generator has been integrated into several variations of an ASAG system. The accuracy of these systems has improved significantly. However, the problem of lack of Arabic language resources is still a challenge for automatic assessment and for any other Arabic TAL activity in general.

Key words: Automatic short answers Grading, Paraphrase, Deep Learning, Dataset

الملخص

يعد إنشاء إعادة الصياغة مشكلة كبيرة في معالجة اللغة الطبيعية، خاصة في أنظمة الأسئلة / الإجابات مثل أنظمة تصنيف الإجابات القصيرة التلقائية (ASAG).

نتناول في عملنا مشكلة التوليد التلقائي لإعادة الصياغة للغة العربية من أجل دمجها في نظام ASAG. اقترحنا ثلاثة نماذج على أساس التعلم العميق. الأول هو نموذج Bi-LSTM يعتبر نموذجًا أساسيًا. والثاني هو جهاز تشفير / وحدة فك ترميز. النموذج الثاني هو جهاز تشفير / وحدة فك ترميز بآلية انتباه أطلقنا عليها اسم EDAM (جهاز فك التشفير مع آلية الانتباه).

أجرينا اختباراتنا باللغتين العربية والإنجليزية باستخدام مجموعتي بيانات، إحداهما باللغة العربية والأخرى باللغة الإنجليزية. تم اختيار الثانية كجزء من تأكيد النتائج التي حصلنا عليها باللغة العربية. يوضح التقييم الكمي للمناهج المقترحة فعالية النموذج الثالث المسمى EDAM.

يؤكد التقييم النوعي البشري على عينة من إعادة الصياغة الناتجة عن نهجنا أن جودة إعادة الصياغة التي تم إنشاؤها جيدة من الناحية اللغوية والنحوية.

تم دمج مولد EDAM الخاص بنا في العديد من الأشكال المختلفة لنظام ASAG. تحسنت دقة هذه الأنظمة بشكل ملحوظ. ومع ذلك، لا تزال مشكلة نقص موارد اللغة العربية تمثل تحديًا للتقييم التلقائي وأي نشاط آخر في معالجة اللغة الطبيعية للغة العربية بشكل عام.

الكلمات المفتاحية: تصنيف الإجابات القصيرة التلقائية، إعادة الصياغة، التعلم العميق، مجموعة البيانات

LISTE DES FIGURES

LISTE DES TABLEAUX

TABLE DES MATIERES

INTRODUCTION GENERALE

L'enseignement en ligne (e-learning) est l'utilisation des nouvelles technologies multimédias de l'Internet, il permet d'améliorer la qualité de l'enseignement en facilitant, d'une part, l'accès à des ressources et à des services, d'autre part, les échanges et la collaboration à distance. Le e-learning fait partie des technologies de l'information et de la communication pour l'éducation (TICE), et permet de réaliser des activités non présentielles. Il a acquis en ces dernières années une grande place dans l'enseignement, voire remplacer l'enseignement classique prochainement. Cependant, quel que soit le type d'enseignement effectué, présentiel ou non, l'intervention de l'enseignant est primordiale pour le bon déroulement des tâches pédagogiques telles-que : la production des ressources pédagogiques, la présentation des cours, le suivi des étudiants ainsi que la correction et l'évaluation des tests et des épreuves. Cette dernière tâche reste très fastidieuse pour l'enseignant, surtout avec un nombre d'étudiants de plus en plus important.

Dans ce souci, plusieurs tentatives pour la mise au point d'un système d'évaluation automatique des réponses ont vu le jour. Plusieurs d'entre eux existent aujourd'hui, majoritairement dédiés aux questions avec réponses fermées comme les questions à choix multiples, vrai ou faux, cases à cocher... Par contre, quand il s'agit des questions ouvertes (exemple qu'est-ce-que la cybercriminalité ?), les systèmes d'évaluation automatique des réponses ne sont pas très efficaces, le taux de corrélation entre l'évaluation automatique et l'évaluation de l'enseignant est relativement moyen à faible.

Ce taux varie d'une approche à une autre selon plusieurs critères comme : le choix de la langue d'évaluation (présence ou non de corpus de données dans telle ou telle langue), l'utilisation ou non des techniques de l'Intelligence Artificielle. D'ailleurs l'IA a montré, jusqu'à présent, son efficacité dans plusieurs domaines y compris les Traitement Automatique de la Langue (TAL). L'évaluation automatique des réponses en fait partie du TAL.

Problématique

Le principe général de l'évaluation consiste à attribuer un score à la réponse de l'étudiant en la comparant avec celle formulée par l'enseignant (réponse modèle). Diverses approches de correction automatique améliorent leur performance en intégrant un nombre important de réponses modèles similaires à chaque question. Ces réponses sont associées à différentes formulations possibles que l'enseignant pourrait imaginer, il s'agit de paraphrases. La mise en place d'un corpus de réponses modèles efficace n'est pas une tâche facile et demande un temps très important. Elle est réalisée souvent manuellement ou par le biais de grammaires que même l'enseignant ne maitriserait pas toujours.

Objectifs

Notre objectif principal consiste, tout d'abord, à automatiser le processus de création d'un corpus de réponses modèles. Nous proposons de générer automatiquement des paraphrases relatives à une réponse modèle formulée par l'enseignant. Une réponse modèle écrite par l'enseignant constitue l'embryon du corpus des réponses modèles. Le reste du corpus sera généré automatiquement. Une approche par dictionnaire distribué a été développée dans le cadre d'un travail précédent. Nous voulons dans ce travail explorer les techniques du Machine Learning et du Deep Learning (l'apprentissage automatique et apprentissage profond) pour améliorer les résultats déjà obtenus.

La génération automatique d'un corpus de réponses modèles par paraphrase doit passer par la réalisation des sous objectifs suivants :

- Explorer les approches de Machine Learning dans la génération de paraphrases.
- Concevoir le modèle de Réseaux de Neurones pour transformer une réponse modèle en entrée en nouvelles réponses ayant le même sens mais formulées différemment (générer les paraphrases)
- Evaluer la qualité du modèle de génération des paraphrases (acquisition de dataset et de résultats de la littérature ainsi que les métriques d'évaluation).
- Intégrer la génération des paraphrases au système d'évaluation automatique.
- Evaluer l'impact des paraphrases sur le système d'évaluation par rapport aux approches déjà développées dans des travaux précédents rentrants dans le même contexte.

Afin d'atteindre nos objectifs fixés, nous structurons notre mémoire en quatre chapitres distincts.

Dans le chapitre 1, nous présentons les concepts fondamentaux du Deep Learning liés à notre problématique.

Le chapitre 2 vise à étudier l'ensemble des travaux et des approches traitant la génération automatique de paraphrases. Nous présentons dans le chapitre suivant, trois modèles basés sur le Deep Learning pour pallier au problème de la génération des paraphrases en langue arabe. Le dernier chapitre porte sur l'intégration du meilleur modèle obtenu au sein de deux systèmes d'évaluation automatique de réponses courtes.

Nous concluons en rappelant les principales contributions et esquisserons des perspectives sur la génération des paraphrases.

CHAPITRE I : CONCEPTS FONDAMENTAUX DU DEEP LEARNING

1- INTRODUCTION

Depuis de nombreuses années maintenant, l'Intelligence Artificielle est appliquée dans plusieurs domaines (Traitement d'image, Robotique, Raisonnement automatique et Acquisition de connaissances, Systèmes décisionnels et Stratégiques, Systèmes Bancaires, bio-informatique, Traitement du langage Naturel, etc.). Son utilité a été largement observée par la communauté scientifique, surtout avec l'émergence du Deep Learning.

Le Deep Learning ou apprentissage profond, est un sous domaine de l'IA dérivé du Machine Learning (Apprentissage automatique). Il s'appuie sur les réseaux de neurones artificiels.

Actuellement, le Deep Learning a montré une grande capacité et efficacité à améliorer les approches du traitement automatique de la langue, telle que la génération des paraphrases, objet de notre travail. Pour cela nous introduisons dans ce chapitre, les concepts fondamentaux de ce nouveau paradigme. Après une brève introduction sur l'IA et le Machine Learning, nous abordons en détail les réseaux de neurones, puis le Deep Learning. Nous nous étalons sur les réseaux de neurones récurrents (RNN) et ses variantes en vue de leur utilisation dans notre travail.

2- INTELLIGENCE ARTIFICIELLE

L'intelligence Artificielle (IA) est l'un des champs les plus récents parmi les sciences et l'ingénierie. Le terme Intelligence Artificielle (IA) a été introduit par McCarthy en 1956 pendant une conférence au Dartmouth College, et depuis, ce terme a été retenu pour représenter le domaine[1].

L'Intelligence Artificielle (IA) est une simulation du processus d'intelligence humaine par des machines comme dans un système informatique. Les systèmes d'IA ont généralement des comportements associés à l'intelligence humaine, tels que la planification, le raisonnement, la

résolution de problèmes, la représentation des connaissances, l'apprentissage, le mouvement, la perception et la manipulation[1].

En 1993, Heer a défini l'Intelligence Artificielle comme étant « l'art de créer des machines capables de prendre en charge des fonctions exigeant de l'intelligence quand elles sont réalisées par des gens » [2].

D'une manière générale, le but de l'IA est de construire des systèmes qui puissent déployer un comportement intelligent, et réaliser des tâches complexes avec un niveau de compétence équivalent, voir supérieur, à celui des humains [3]

L'IA est directement liée aux concepts de Systèmes à base de Connaissances, Systèmes Experts, Systèmes Intelligents, Acquisition de Connaissances, et l'Apprentissage Automatique. Ce dernier est considéré comme une des branches de l'Intelligence Artificielle les plus importantes, et l'une des principales caractéristiques des systèmes intelligents à qui nous consacrons la section suivante pour le définir.

3- MACHINE LEARNING

L'apprentissage automatique (Machine Learning en anglais), est l'un des sous domaines de l'IA. Il a pour objectif d'extraire et d'exploiter automatiquement l'information présente dans un jeu de données. Il couvre un vaste champ d'objectifs comme la fouille de données, la classification, la sélection de variables, la discrimination, la régression, la sélection de modèle, la génération et l'inférence de règles, le traitement automatique de la langue, etc.

Une des définitions du Machine Learning est donnée par H. Simon comme suit: « L'apprentissage dans un système est indiqué par les changements qu'il subit. Ces changements sont adaptatifs dans le sens où ils rendent possible au système de réaliser une même tâche, ou des tâches tirées d'une même population, d'une façon plus efficace et plus efficiente la prochaine fois qu'elle sera réalisée » [4]. En d'autres termes, « le Machine Learning est réalisé par des outils permettant d'acquérir, élargir et améliorer les connaissances disponibles au système. En général, l'apprentissage implique des processus d'adaptation et de modification des structures de contrôle et/ou de représentation de connaissances du système en question » [3].

L'apprentissage automatique se présente aujourd'hui comme une des meilleures alternatives pour améliorer les processus d'acquisition des connaissances, surtout avec l'émergence de l'Apprentissage profond (Deep Learning, ce dernier est une variante des Réseaux de neurones,

une des différentes méthodes de l'Apprentissage Automatique. Nous détaillerons ces notions dans les sections suivantes. La Figure I.1 comportant un schéma distinguant entre les différentes notions de l'IA.

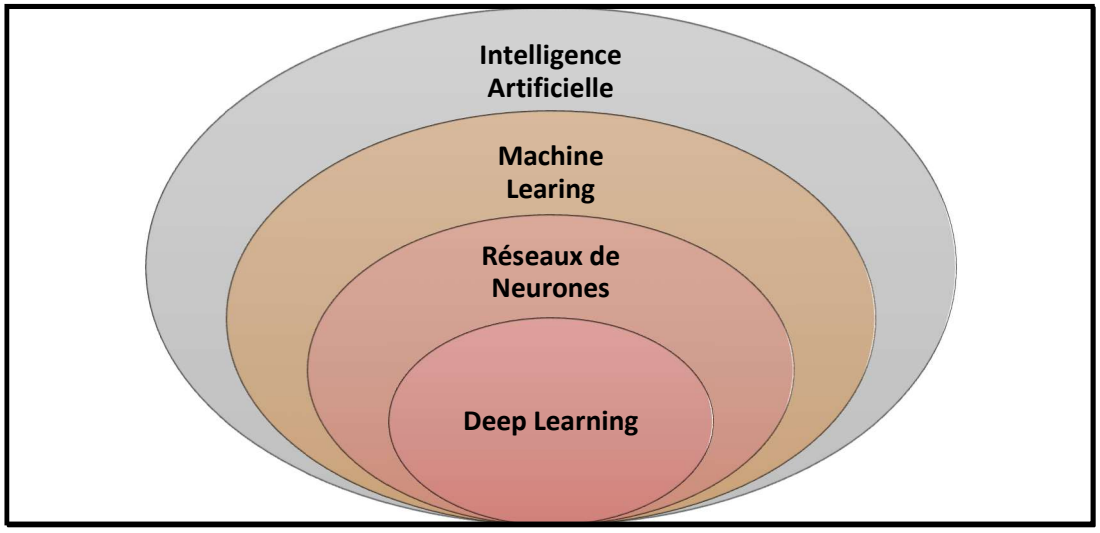


Figure I-1-IA, ML, NN et Deep Learning [5]

4- RESEAUX DE NEURONES (NEURAL NETWORKS 'NN')

Inspiré du neurone biologique, le réseau de neurone est un ensemble de neurones interconnectés, il est conçu pour aider l'utilisateur à résoudre un problème généralement complexe, comme la reconnaissance d'images et le traitement automatique du langage naturel[6].

4-1 Fonctionnement du réseau de neurones

Un réseau de neurones fonctionne de manière similaire au réseau de neurones du cerveau humain. Il est composé d'une suite de couches dont la première couche correspond à la couche d'entrée (input layer), et la dernière est celle de la sortie (output layer).

Chaque couche est composée de neurones, leurs sorties représentent l'entrée de la couche suivante. Ceci est illustré dans la Figure I.2.

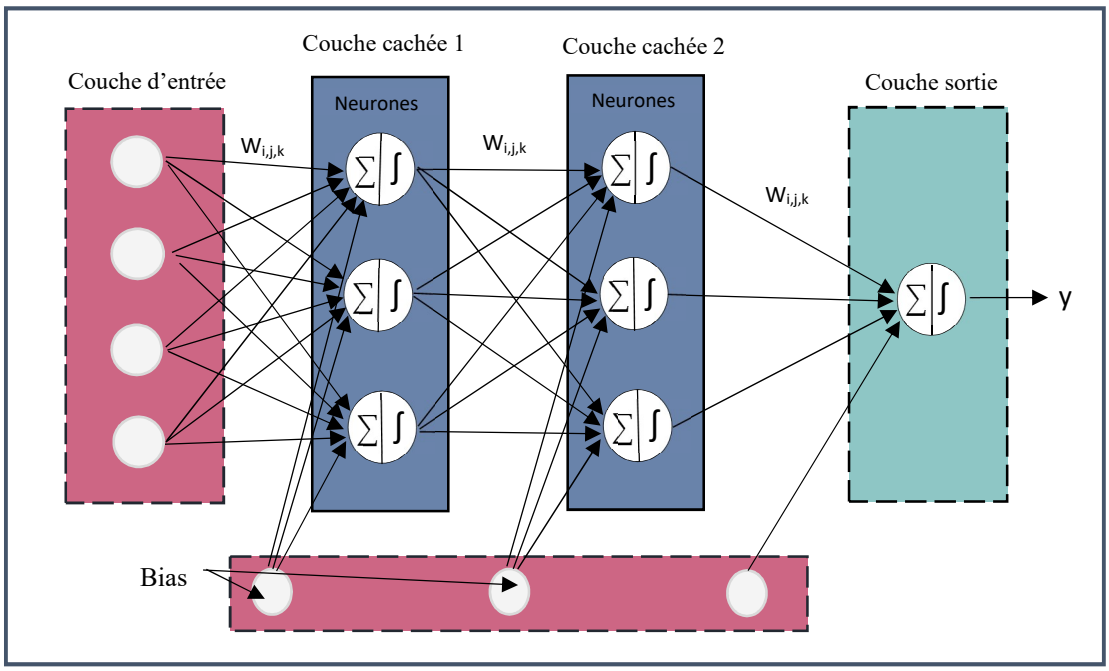


Figure I-2 Architecture d'un réseau de neurones NN [7], [8]

4-2 Les fonctions d'activation

Les fonctions d'activation sont utilisées afin d'ajuster la valeur d'une sortie d'un neurone(Y). Cette valeur est comprise entre $-\infty$ et$+\infty$, elle est calculée par l'équation :

$$Y = \sum(weight * input) + bias$$

Où le biais est un paramètre supplémentaire dans le réseau neuronal, il est utilisé pour ajuster la sortie avec la somme pondérée des entrées du neurone. Ainsi, le biais est une constante qui aide le modèle à s'adapter le mieux aux données entrantes.

La fonction d'activation permet donc de limiter la valeur de la sortie (Y), et ainsi de décider si le neurone doit être activé ou non. On distingue plusieurs types de fonctions d'activations.

4-2-1 Fonction d'activation binaire [9], [10]

Consiste à prendre en considération une valeur du seuil '***S'***. Dans ce cas si: ***Y*** est supérieur à ***S***, alors le neurone est activé. Le graphe de cette fonction est illustré dans la Figure I.3.

$f(Y) = 1 \quad si\ Y \geq S$

$f(Y) = 0 \quad si\ Y < S$

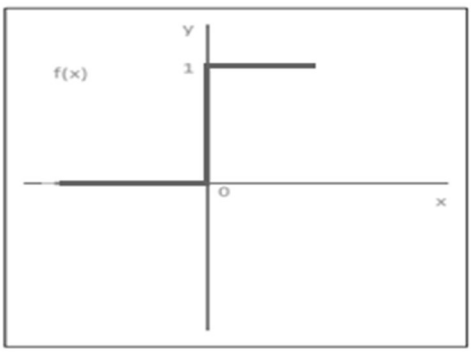


Figure I-3 Représentation graphique de la fonction binaire

4-2-2 Fonction d'activation linéaire [9], [10]

Il s'agit d'une simple fonction d'activation en ligne droite. La Figure I.4 montre la représentation graphique de la fonction linéaire. Cette fonction produit une sortie qui est proportionnelle à l'entrée. La sortie est la somme pondérée des entrées. Ceci est de la forme :

$$f(z) = z$$

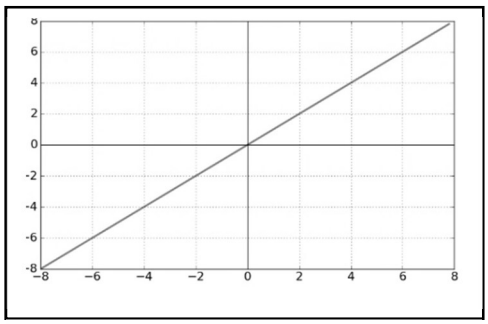


Figure I-4 Représentation graphique de la fonction d'activation linéaire

Les fonctions binaires et linéaires régissent dans une situation de proportionnalité qui sont des cas particuliers, pour le cas général, il faudra utiliser des fonctions non-linéaires. Ces dernières permettent de créer des transformations complexes entre les entrées de manière à obtenir une valeur de sortie.

4-2-3 Fonctions d'activation non linéaire

Les fonctions non linéaires permettent de séparer les données non linéairement séparables. Elles constituent les fonctions d'activation les plus utilisées. Une équation non linéaire régit la correspondance entre les entrées et la sortie. Nous trouvons dans la littérature plusieurs fonctions d'activation non linéaires, qui sont:

- **Sigmoïde** [9], [10]

La fonction sigmoïde est largement utilisée, notamment dans les modèles où nous devons **prédire la probabilité** en sortie, étant donné sa valeur qui est comprise entre 0 et 1. La Figure I-5 représente le graphe de cette fonction.

$$g(z) = \frac{1}{1+e^{-z}}\ , z \in \mathrm{R}$$

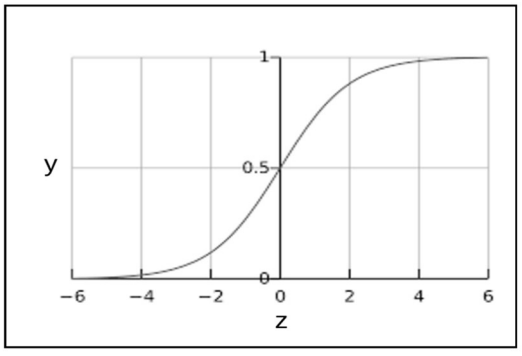


Figure I-5 Représentation graphique de la fonction sigmoïde

- **Fonction Tanh (tangente hyperbolique)** [9], [10]

La plage de la fonction tanh (illustrée dans la Figure I.6) est comprise entre (-1 et 1). Elle est également très populaire.

$$f(x) = \tanh(x) = \frac{2}{1+e^{-2x}} - 1$$

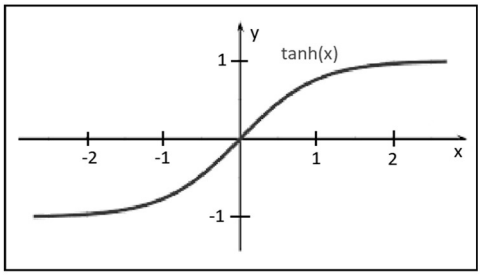


Figure I-6 Représentation graphique de la fonction Tanh

Notons ici que ces deux fonctions souffrent du problème de la disparition du gradient, i.e. lors de l'étape de la rétro propagation, les gradients ont tendance à devenir de plus en plus petit, ce qui rend l'apprentissage beaucoup plus lent.

- **Fonction RELU (Rectified Linear Unit)** [9], [10]

La fonction RELU représentée dans la Figure I.7, est couramment utilisée, elle propose une solution au problème de disparition du gradient, mais elle peut aussi conduire à l'explosion du gradient à cause de l'infinité des valeurs qu'elle peut prendre. De ce fait, cette fonction est considérée comme inappropriée pour les réseaux de neurones récurrents.

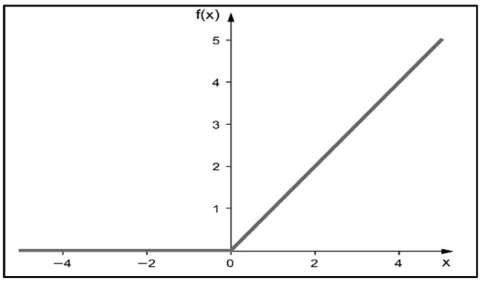


Figure I-7 Représentation graphique de la fonction RELU

$$f(z) = 0 \; si \; z < 0$$

$$f(z) = z \; si \; z >= 0$$

Cependant grâce aux avancées technologiques, et à des bases de données toujours plus conséquentes aujourd'hui, les couches de neurones ont pu prendre du volume et ainsi passer d'un apprentissage passif à un apprentissage dynamique avec ce qu'on appelle le Deep Learning ou l'Apprentissage profond.

- **Fonction Softmax** [9], [10]

La fonction softmax est une fonction qui permet de normaliser un vecteur de nombres réels en un vecteur de nombre compris entre 0 et 1. Elle suit une distribution de probabilité dont le total est égal à 1. La fonction softmax est définie par la formule :

$$S(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \qquad pour \; j \in \{1, \ldots, K\}$$

Où:

z_j: Sont les éléments du vecteur d'entrée et peuvent prendre n'importe quelle valeur réelle.

K: Le nombre de classes dans le classificateur multi-classes.

5- DEEP LEARNING

Le Deep Learning (Apprentissage Automatique Approfondi) est une nouvelle approche du Machine Learning et une révolution des Réseaux de Neurones. Il se base sur l'apprentissage en plusieurs couches de représentations hiérarchique de concepts. Chaque nœud d'une couche de niveau supérieur est défini en se basant sur les nœuds de la couche inférieure. De même les nœuds d'une couche, peuvent aider dans la définition de plusieurs autres concepts de la couche supérieure [11], [12].

Plusieurs techniques du Deep Learning ont été mises en place. Elles sont classées en deux grandes catégories, les réseaux de neurones convolutionnels utilisés principalement avec le traitement d'images et les réseaux de neurones récurrents utilisés principalement dans le TAL. Nous nous intéressons plus particulièrement à la seconde catégorie que nous détaillons dans cette section.

5-1 Modèles de séquences

Les modèles de séquences sont utilisés dans une variété d'applications. Ils peuvent traiter tout type de problème de prédiction contenant une série chronologique dans les couches d'entrée ou de sortie. Nous citons quelques exemples d'utilisation :

- Analyse d'une séquence ADN : l'extraction des caractéristiques d'une séquence ADN.
- Reconnaissance de l'activité vidéo : la détection de certaines caractéristiques d'une séquence vidéo.
- La reconnaissance vocale (Speech To Text) : qui permet d'analyser une entrée de type audio et la transcrire sous forme d'un texte.
- La génération du son (Text To Speech) : Générer une voix à partir d'une séquence de texte.
- Reconnaissance d'entité de nom : détecter les noms qui appartiennent à une certaine catégorie (nom propres, noms d'animaux, etc.) à partir d'un texte.
- La traduction : consiste à traiter un texte en entrée et générer sa traduction dans une

autre langue…

Les modèles de séquence ont prouvé leur efficacité dans le domaine du Deep Learning, et ce, lorsque l'entrée ou la sortie est une séquence de données. De ce fait, nous nous intéressons dans notre travail à ces modèles et plus particulièrement les RNN, les LSTM, les Encodeurs/Décodeurs ainsi que leurs variantes.

5-2 Réseaux de neurones récurrents

Les réseaux de neurones récurrents (RNN) [13] constituent une famille de réseaux de neurones pour le traitement des données séquentielles. Ce type de réseaux accepte des entrées de taille variable, génère des résultats en prenant en compte les relations entre les séquences (Figure I-7). L'architecture standard d'un RNN est illustrée dans la figure I.8.

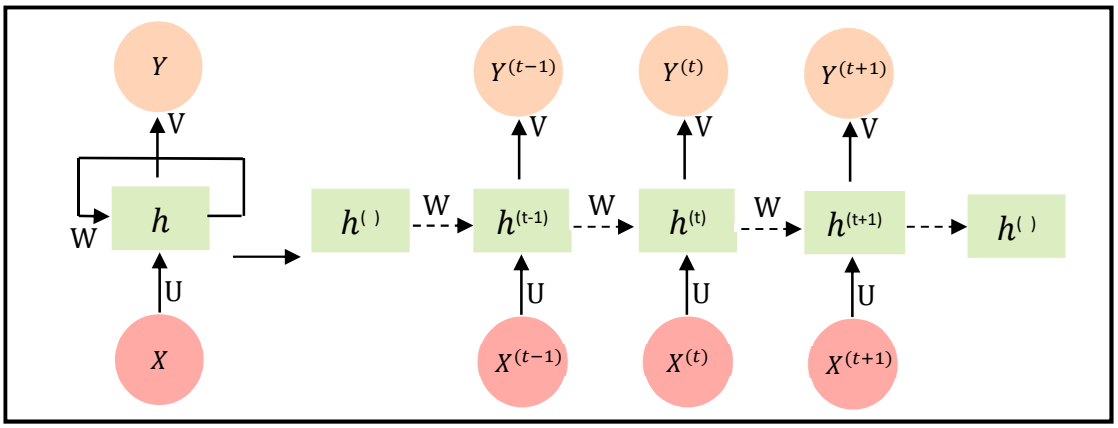


Figure I-8 Architecture d'un réseau de neurones récurrent standard [14]

Comme le montre la figure ci-dessus, les entrées sont liées les unes aux autres. La valeur de la fonction $h(t)$ dépend de la valeur de $h(t-1)$ et de l'entrée $x(t)$.

Plusieurs architectures des réseaux de neurones récurrents existent, le choix de d'une ou de l'autre dépend essentiellement de la nature du problème qu'on veut résoudre. La figure I.9 ci-dessous montre les différentes architectures existantes.

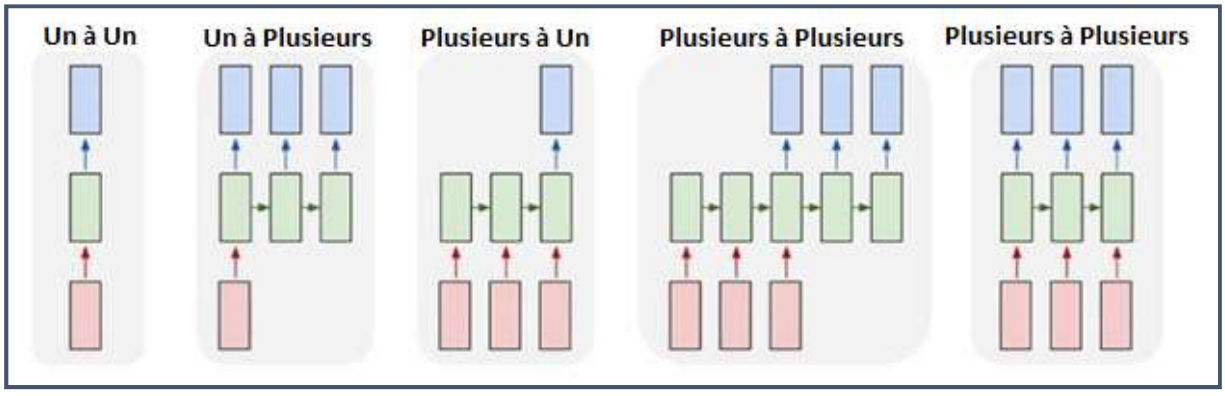


Figure I-9 Les différentes architectures de RNN [14]

Le rectangle rouge représente les entrées du réseau (scalaire ou vecteur), le vert représente la cellule du RNN, et celui en bleu, les sorties (pouvant être un scalaire ou un vecteur).

- **Un à Un**

C'est la représentation la plus simple d'un RNN, avec une seule entrée et une seule sortie. Par exemple, la classification d'images.

- **Un à Plusieurs**

Ce modèle génère une séquence à partir d'une seule entrée. Par exemple la génération d'une phrase en fonction d'une entrée qui serait le premier mot.

- **Plusieurs à Un**

Cette architecture permet de générer une seule sortie (classification) à partir d'une séquence d'entrée. Ce modèle est utilisé dans diverses activités telles que la classification des sentiments.

- **Plusieurs à Plusieurs**

Cette architecture peut être représentée selon deux manières différentes.

- La première méthode consiste à générer une séquence après la lecture de toutes les entrées, la traduction entre langues est un exemple qui utilise cette représentation.
- La deuxième méthode consiste à générer des sorties au fur et à mesure en fonction d'entrées. Un exemple concret est la génération des labels à partir de chaque image d'une vidéo.

Malgré les performances présentées par les réseaux de neurones récurrents, néanmoins ils possèdent certaines limites :

- Modèle à mémoire courte : Il ne prend en considération que les séquences proches.
- Temps de traitement très important.
- Le modèle ne prend pas en considération les séquences ultérieures à une cellule donnée, ce qui est un handicap dans le TAL.

5-3 Les réseaux de neurones récurrents bidirectionnels

Comme mentionné précédemment, l'une des limites des RNN standards est qu'ils ne permettent pas de prendre en compte les informations futures à un état donné.

Les réseaux de neurones récurrents bidirectionnels (BRNN) offrent une solution à ce problème. L'architecture BRNN est illustrée dans la Figure I.10.

Exemple :

« **Blanc** Laurent est un ancien joueur de l'équipe nationale française de football »

« En synthèse additive des couleurs, le **blanc** s'obtient par un mélange équilibré des trois couleurs primaires. »

Dans cet exemple, les mots qui précèdent « blanc » ne suffisent pas pour déterminer son vrai sens, d'où la nécessité de prendre en compte les mots qui le suivent.

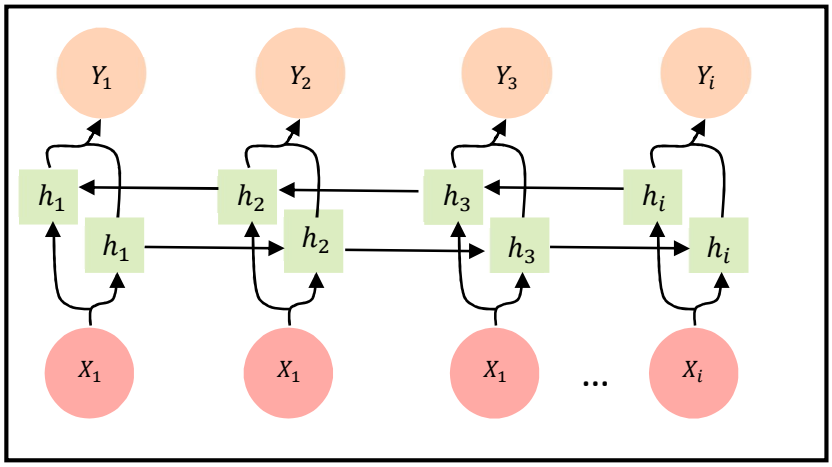


Figure I-10 Architecture d'un BRNN [15]

Le principe est d'appliquer la propagation* vers l'avant 2 fois, une pour les cellules avant et une pour les cellules arrière. Les deux activations (avant, arrière) seraient considérées pour calculer la sortie $\hat{y}$ au temps t.

L'un des avantages majeurs des BRNN est qu'ils permettent la prise en considération des données ultérieures afin de prédire une valeur à un instant t. Mais ce modèle arrive à ses limites lorsqu'il s'agit de traiter des séquences de taille énorme, on parle du problème de disparition de gradient.

Exemple :

La *fille qui mangeait du pain, et buvait du lait, était* ***brune****.*

Le *garçon qui mangeait du pain, et buvait du lait, était* ***brun****.*

Dans l'exemple ci-dessus, la distance entre les deux mots 'le' et 'brun' constitue un réel problème, ce qui ne permet pas à l'algorithme de détecter la relation entre eux (mémoire courte). Un autre souci que l'on rencontre lors de l'étape de la rétro propagation dans le cadre des RNN possédant un nombre de couches élevé est la diminution rapide de la valeur du gradient, ce fait

* La rétro propagation du gradient est une méthode statistique pour calculer le gradient de l'erreur pour chaque neurone d'un réseau de neurones, de la dernière couche vers la première.

mène à la négligence des mots qui se trouvent au début de la phrase.

Pour les réseaux peu profonds avec seulement quelques couches, cela ne pose pas un réel problème [16]. Cependant, lorsque plusieurs couches sont utilisées, la valeur de gradient converge rapidement vers zéro, ce qui rend l'entrainement moins performant. En effet, lors de l'étape de la rétro propagation, une dérivée partielle est calculée de la couche finale à la couche initiale afin de minimiser l'erreur, la convergence rapide de cette dérivée vers zéro engendre le problème de mémoire courte appelé la disparition du gradient (Vaniching Gradient).
Un autre problème de l'entraînement des réseaux de neurones se pose, c'est l'explosion du gradient, à l'inverse du premier problème, ici, lors de l'entrainement d'un réseau très profond, les dérivées ou les pentes peuvent devenir très grands, ceci rend l'entraînement difficile, ce phénomène est appelé l'explosion du gradient (Explosing Gradient).

Pour remédier à ces problèmes, de nouvelles variantes du RNN ont vu le jour telles que les LSTM et les GRU, nous les détaillerons dans les sections suivantes.

5-4 Long Short-Term Memory (LSTM)

Les réseaux de mémoire court et long terme [17] est une version améliorée de RNN, ils permettent de prendre en charge le lien entre les mots distants. Les LSTM ont été introduits par Hochreiter et Schmidhuber en 1997. De nos jours, ils sont largement utilisés dans la résolution de divers problèmes (Figure I.11).

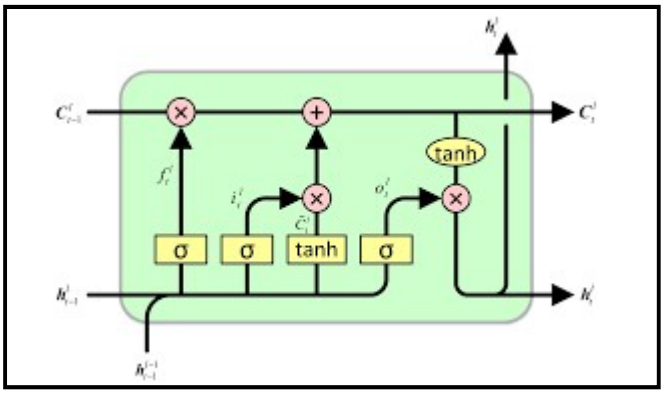


Figure I-11 Architecture détaillée d'une cellule LSTM [18]

Contrairement aux RNNs classiques, les LSTM proposent une solution au problème de disparition du gradient en introduisant un mécanisme de portes (gates) et une cellule mémoire, celles-ci sont représentées dans la Figure I-11. Ces portes contrôlent le flux d'informations du

réseau. Quant à la cellule mémoire, elle permet de garder un état aussi longtemps que nécessaire.

Ce réseau prend en entré $h_{(t-1)}$ qui est la sortie de la précédente unité LSTM, $X(t)$ l'entrée du pas de temps actuel et $C_{(t-1)}$ la « mémoire » de l'unité précédente.

Les LSTM ont la capacité de supprimer ou d'ajouter des informations à l'état de la cellule, soigneusement régulées par les portes. Ces dernières sont un moyen de laisser éventuellement passer l'information. On distingue trois types de portes:

5-4-1 Porte d'oubli (Forget gate)

En examinant la valeur de sortie prédite par la couche LSTM précédente h_{t-1} ainsi que l'entrée x_t, la porte décide des informations qui doivent être omises de la cellule mémoire précédente C_{t-1}. Pour filtrer ces valeurs nous allons utiliser la fonction suivante :

$$f_t = \sigma\big(W_f[h_{t-1}, x_t] + b_f\big)$$

Où, $[h_{t-1}, x_t]$ correspond à la concaténation des deux vecteurs h_{t-1} et x_t, W_f correspond au poids des neurones et b_f au biais**.**

f_t retourne un vecteur dont les valeurs sont comprises entre 0 et 1, cela est dû à l'utilisation de la fonction sigmoïde. Ces valeurs représentent le degré d'importance de chaque information présente dans la cellule précédente c_{t-1}

Une multiplication terme à terme est appliquée entre f_t et c_{t-1}.

$$C' = f_t * c_{t-1}$$

Ainsi lors de cette multiplication, les valeurs proches de 0 dans C' ne seront pas autorisées à passer, à l'inverse, si les valeurs sont proches de 1 les informations seront préservées et autorisées à passer. La représentation de la porte d'oubli est illustrée dans la Figure I.12.

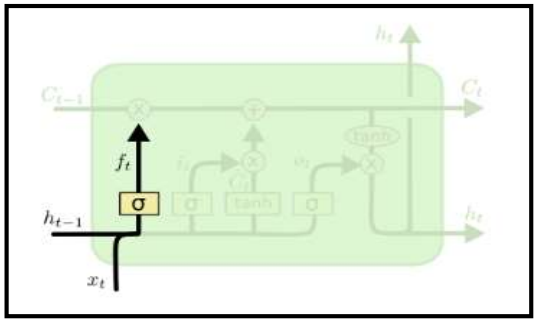


Figure I-12 Représentation de la porte d'oubli [18]

5-4-2 Porte d'entrée (Input gate)

La porte d'entrée représentée dans la Figure I.13, permet d'ajouter de nouvelles informations en se basant principalement sur les informations provenant de l'instant t à savoir x_t. Pour ce faire, deux fonctions mathématiques sont utilisées :

$$\check{C}_t = \tanh(W_c[h_{t-1}, x_t] + b_c)$$

Dont le but est de proposer de nouvelles informations dites informations candidates qui peuvent être misent dans la mémoire. Cette fonction génère un vecteur de valeurs comprises entre -1et1. Quant à la deuxième opération i_t, son but est de sélectionner à partir de C'_t les informations susceptibles d'être ajoutées à la cellule existante C.

$$i_t = \sigma\,(W_i[h_{t-1}, x_t] + b_i)$$

Le vecteur résultant de la multiplication de C'_t et i' définit l'état de la porte d'entrée.

Cet état représente les informations pertinentes qu'on veut garder afin qu'elles soient réutilisées dans la mémoire. L'état de cette dernière est calculé comme suit:

$$C_t = f_t * C_{t-1} + i_t * \check{C}_t$$

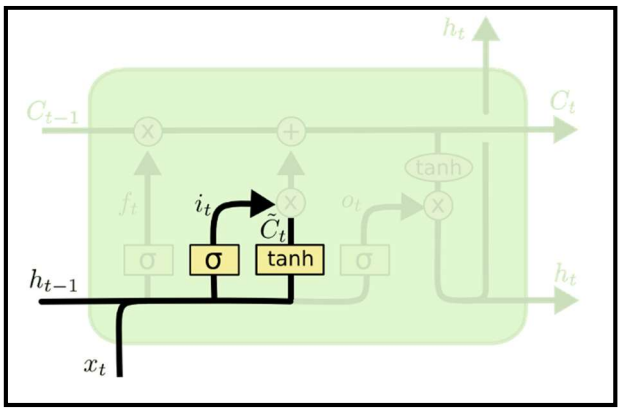


Figure I-13 Représentation de la porte d'entrée [18]

5-4-3 Porte de sortie (output gate)

Le but est de définir la valeur de sortie de la cellule à un instant t.

Dans un premier temps une fonction sigmoïde est appliquée afin de sélectionner les valeurs qui seront autorisées à passer.

$$O_t = \sigma\,(W_o[h_{t-1}, x_t] + b_o)$$

Dans un second temps, le vecteur obtenu de l'étape précédente ainsi que l'état de mémoire C_t seront utilisés afin de définir la sortie h_t de la cellule en question à l'instant t.

$$h_t = O_t * \tanh(C_t)$$

L'architecture de cette porte est illustrée dans la Figure I.14.

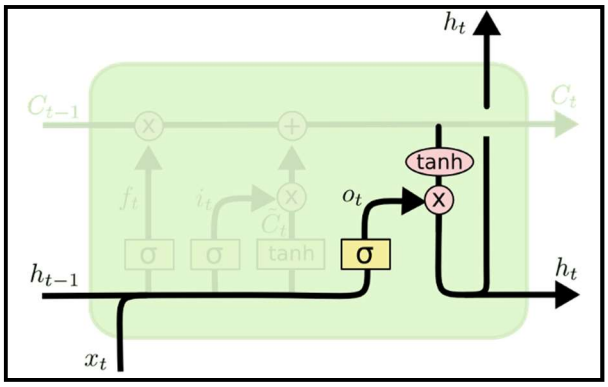


Figure I-14 Représentation de la porte de sortie [18]

5-5 Gated recurrent unit (GRU)

Semblable aux LSTMs, les réseaux de neurones récurrents à portes (GRU), illustrés dans la Figure I.15 offrent une solution au problème du gradient disparaissant. Ils possèdent deux portes, une pour la réinitialisation et une autre pour la mise à jour. Ils utilisent aussi un mécanisme d'état caché, contrairement aux LSTMs qui utilisent un état de cellule et 3 portes.

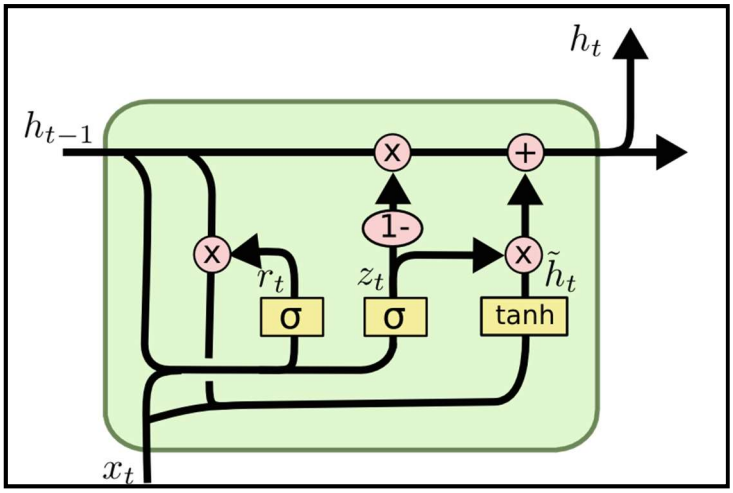


Figure I-15 Architecture d'une cellule GRU [18]

5-5-1 Porte de mise à jour

A partir des états passés, la porte de mise à jour détermine la quantité d'informations devant être transmises dans le futur. Elle est calculée comme suit:

$$z_t = \sigma\left(W_z * x_t + U_z * h_{t-1}\right)$$

Où W_z et U_z représentent respectivement les poids de x_t et h_{t-1}.

5-5-2 Porte de réinitialisation

Cette porte est utilisée pour décider de la quantité d'informations du passé à oublier. Celle-ci est calculée par la formule suivante:

$$r_t = \sigma(W_r.x_t + U_r.h_{t-1})$$

Comme pour z_t, W_r et U_r représentent les poids de x_t et h_{t-1}.

5-5-3 Contenu actuel de la mémoire

Les GRU introduisent un nouveau contenu de mémoire ($\hat{h}_t$) qui utilisera la porte de réinitialisation pour stocker les informations pertinentes du passé. Ce dernier est calculé comme suit:

$$\hat{h_t} = \tanh(W.x_t + r_t \odot U.h_{t-1})$$

Où $r_t \odot U.h_{t-1}$ est la multiplication terme à terme (produit de Hadamard) entre les deux vecteurs.

5-5-4 Mémoire finale au pas de temps actuel

Afin de transmettre l'information de la cellule actuelle aux autres unités du réseau, il est nécessaire de prendre en considération la porte de mise à jour pour déterminer ce qu'il faut collecter à partir du contenu actuel de la mémoire $\hat{h}_t$ et des étapes précédentes h_{t-1}. Cela se fait comme suit:

$$h_t = z_t \odot h_{t-1} - (1 - z_t) \odot \hat{h_t}$$

5-6 Les réseaux de neurones LSTM bi directionnels (Bi LSTM)

Tout comme les RNN bidirectionnels, les Bi LSTM utilisent une double propagation vers l'avant en utilisant des cellules LSTM. La sortie $\hat{y}$ est calculée comme suit:

$$\hat{y}^{<t>} = g(W_y [\vec{a}^{<t>}, \overleftarrow{a}^{<t>}] + b_y)$$

L'architecture générale d'un réseau de neurones Bi LSTM est illustrée dans la Figure I.16.

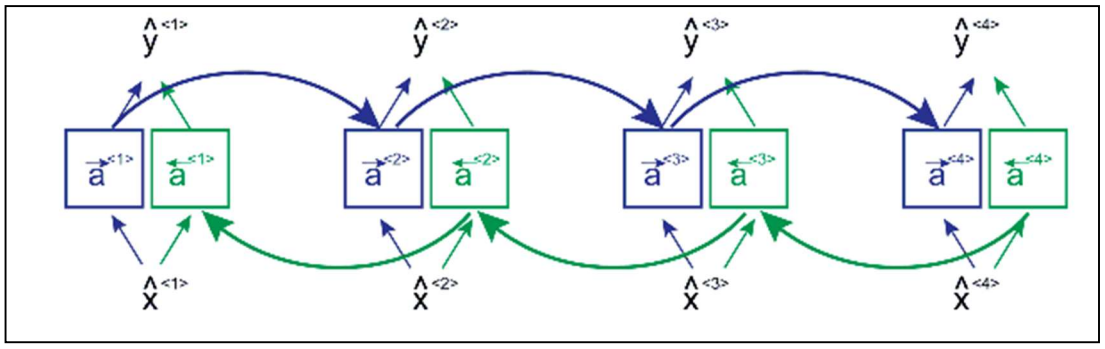


Figure I-16 L'architecture générale d'un réseau de neurones Bi LSTM [19]

Tous les modèles à séquences cités jusqu'à présent sont basés sur des architectures permettant le mappage entre des entrées sorties de taille fixe (un à un, un à plusieurs, plusieurs à un), aucun ne traite le problème de génération à partir d'une séquence à une autre séquence (plusieurs à plusieurs).

Ce type d'architecture est pris en charge par le modèle encodeur/décodeur, qui lui est capable de générer une séquence cible à partir d'un séquence source [20].

5-7 Encodeur-Décodeur

L'encodeur-décodeur est un modèle de conception de réseau neuronal qui permet de générer une sortie de séquence pour une entrée de séquence.

Comme le montre la figure I-17, l'architecture est composée de deux parties, l'encodeur et le décodeur, chaque partie utilise des réseaux de neurones approfondis, plus fréquemment des réseaux neuronaux récurrents (RNN) afin de gérer les entrées de séquence de longueur variable.

Les principaux avantages de cette approche sont :

- La possibilité de former un modèle de bout en bout unique (directement sur les phrases source et cible).
- La capacité de gérer des séquences de texte d'entrée et de sortie de longueur variable.

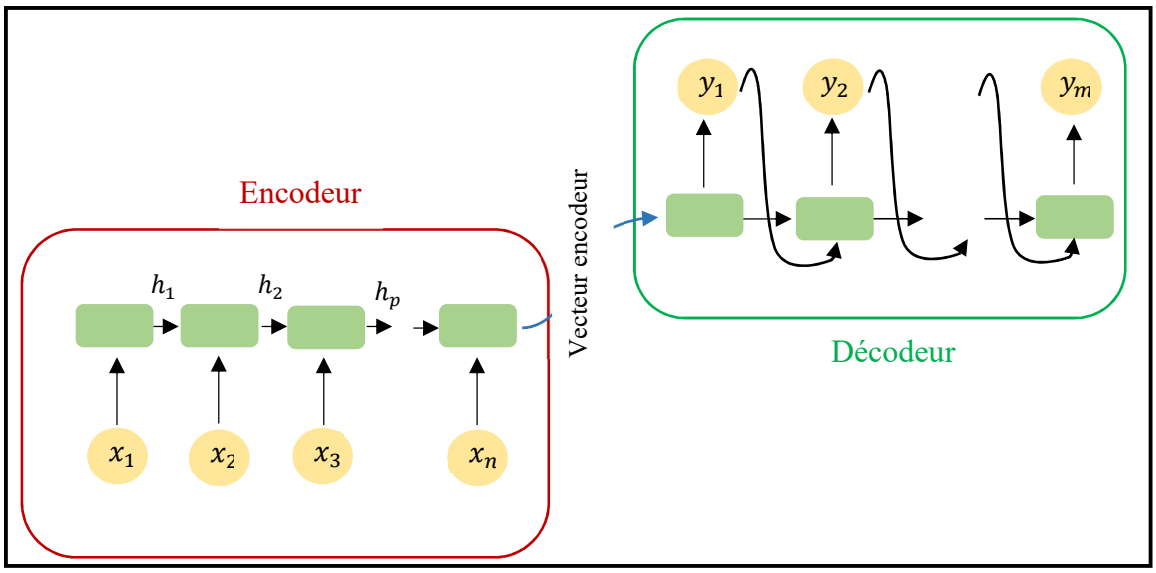


Figure I-17 Architecture du modèle Encodeur-Décodeur [21]

Chaque unité de l'encodeur analyse l'entrée (x_i) et collecte un ensemble d'informations, il sera

envoyé par la suite, à l'unité suivante (propagation vers l'avant). Ensuite, l'état caché final de l'encodeur (vecteur encodeur) est transmis directement au décodeur.

Le décodeur génère la première sortie (y_1) en analysant le vecteur encodeur. Chaque sortie y au pas de temps $t > 1$ dépend de l'état caché et la sortie à $t - 1$

Exemple :
Dans la génération de paraphrases, l'encodeur transforme une phrase source, par exemple, «Hello word», en un état, qui capture ses informations sémantiques. Le décodeur utilise ensuite cet état pour générer la phrase cible traduite, par exemple « Hi everyone ».

6- MECANISME D'ATTENTION

Le mécanisme d'attention fait l'objet d'une technique utilisée dans les réseaux neurones, et plus précisément dans les tâches du traitement automatique de la langue. L'objectif derrière l'utilisation de ce mécanisme est de se focaliser sur certains facteurs pouvant influencer sur la qualité du modèle. La contribution majeure de ce mécanisme est, d'améliorer la performance des modèles « sequence to sequence » (codeur-décodeur) [22], [23].

Le mécanisme d'attention gère et quantifie l'interdépendance dans les éléments d'entrée (Self-Attention) et entre les entrées et les sorties (General Attention). Ce mécanisme a été introduit pour régler l'un des problèmes des modèles « sequence to sequence » à savoir leur incapacité à fournir de bons résultats lorsqu'il s'agit de séquences de taille longue. Ce problème réside au niveau du décodeur où seul le dernier état caché généré par l'encodeur est utilisé comme un vecteur de contexte.

Cette limite est prise en charge par le mécanisme d'attention, ce dernier utilise tous les états cachés de l'encodeur pour générer un vecteur de contexte à chaque pas de temps. Il calcule des scores d'alignement entre l'état caché précédent du décodeur et tous les états cachés de l'encodeur, par la suite la fonction softmax est appliquée sur les scores d'alignement après les avoir représentés sous forme de vecteur. Ses derniers sont multipliés par les états cachés de l'encodeur pour créer le vecteur de contexte. Le vecteur de contexte, l'état caché précédent du décodeur ainsi que la sortie du précédent décodeur sont concaténés pour l'alimentation du décodeur du pas de temps actuel pour produire la sortie.

7- CONCLUSION

Dans ce chapitre nous avons expliqué les notions de base liées à l'Intelligence Artificielle, l'apprentissage automatique, les réseaux neurones, et les réseaux de neurones approfondis. Ce dernier aspect est considéré comme étant la plus récente avancée technologique de l'IA.

Notre travail s'articule sur le Traitement Automatique de la langue, et plus précisément la génération automatique de paraphrases, nous cherchons à proposer une solution plus performante à cet aspect en incluant les approches du Deep Learning.

Nous présentons dans le prochain chapitre les différentes approches de génération de paraphrases, les techniques utilisées et comparer tous les travaux connexes qui se sont basés sur les réseaux de neurones approfondis dans le chapitre suivant.

CHAPITRE II : ETAT DE L'ART GENERATION DE PARAPHRASES

1- INTRODUCTION

La diversité linguistique retrouvée dans les langues en usage comporte un grand défi pour la majorité des applications du TAL [24], Par exemple, elle peut se manifester dans le fait qu'une même idée, un même concept, ou un même évènement peut être exprimé avec des mots complètement différents mais ayant la même signification dans un même contexte, ce groupe de mots représente ce qu'on appelle des paraphrases.

Capturer ou générer automatiquement des équivalences sémantiques (des paraphrases) entre des unités est une tâche complexe mais qui s'avère indispensable dans de nombreux contextes. Dans ce chapitre, nous allons, tout d'abord, commencer à introduire la notion de paraphrase, ses différentes classifications, ainsi que les approches classiques du TAL pour la génération de paraphrases. Nous étudions, ensuite, quelques travaux traitant la génération de paraphrases. Nous présenterons, après, les différents corpus (datasets) utilisés ainsi que les métriques usuelles d'évaluation. Nous terminerons notre chapitre par une comparaison et une synthèse.

2- TRAITEMENT AUTOMATIQUE DU LANGAGE NATUREL

« Le Traitement du Langage Naturel (TAL) est une gamme de techniques de calcul basée sur la théorie pour l'analyse automatique et la représentation du langage humain. La recherche en TAL a évolué de l'ère des cartes perforées et du traitement par lots, où l'analyse d'une phrase peut prendre jusqu'à 7 minutes, à l'ère de Google et autres, où des millions de pages Web peuvent être traitées en moins d'une seconde. Le TAL permet aux ordinateurs d'effectuer un large éventail de tâches liées au langage naturel à tous les niveaux, allant de l'analyse syntaxique et du balisage de partie de la parole (POS pour Part Of Speech tagging) à la traduction automatique et aux systèmes de dialogue. » [25]

Contrairement à l'humain, l'ordinateur trouve des difficultés à comprendre l'intention sous-jacente derrière les phrases. Cette dernière rencontre des ambiguïtés au niveau des mots, phrases

et sens.

Le TAL est utilisé pour résoudre différents types de problèmes qui traitent du texte, tel que l'évaluation des réponses courtes.

3- LA PARAPHRASE

La paraphrase consiste à réécrire une phrase en utilisant des termes ou des mots différents, et ce, en respectant le fait qu'elles aient le même sens, par exemple, « Quelle est la distance entre Alger et Blida » avec « Combien de kilomètres il y a entre Alger et Blida ».

Produire ou générer des paraphrases est le fait de construire des phrases à partir d'autres déjà données afin de simplifier un texte, d'en faire un résumé, d'en faire une analyse sémantique ou reformuler des recherches web, etc.

Cependant, la complexité du langage naturel implique que la génération automatique des paraphrases est une tâche très difficile et complexe. [26], [27].

3-1 Classification des paraphrases

Les paraphrases peuvent être classifiées selon deux critères: le niveau de granularité et le niveau d'analyse de la langue.[28]

3-1-1 Niveau de granularité

Le niveau de granularité des paraphrases concerne la taille des séquences. Les séquences à un seul mot sont appelées paraphrases lexicales, les phrases qui partagent la même signification sont appelés des paraphrases phrastiques. Quant aux fragments de phrases exprimant le même sens sont nommés paraphrases sous phrastiques.[28]

a- Paraphrase lexicale

Ce type concerne les unités lexicales individuelles ayant un sens similaire. Ces unités peuvent avoir une relation de synonymie tel que (désignation - appellation) ou bien une relation d'hyperonymie qui présente la notion de spécification/généralisation comme (fruit - pomme).

b- Paraphrase sous-phrastique

Contrairement aux paraphrases lexicales, les paraphrases sous phrastiques recouvrent des fragments de texte (groupes de mots) présentant la même signification comme (« réussira-t-elle

» – « a-t-elle fait un succès ») ou bien («X ne contredit pas Y » - «X est en accord avec Y »).

c- Paraphrase phrastique

Les paraphrases phrastiques sont des phrases qui transmettent le même sens en changeant seulement quelques mots et/ou passages comme dans cet exemple (« elle a accepté ses excuses en souriant » - « elle a souri lorsqu'il lui avait demandé pardon »).

3-1-2 Niveau d'analyse de la langue

La distinction des différents types de paraphrases au niveau d'analyse de la langue est basée sur la compréhension de la phrase et la détermination des conditions nécessaires pour cette tâche.

Nous distinguons en effet, deux notions à savoir l'énoncé et la phrase. La phrase est un ensemble stable et constant de composants structurés exprimant une idée bien déterminée indépendamment du contexte, alors que si la phrase est considérée vraie ou fausse selon un contexte (circonstances, lieu, moment) et un co-contexte (son entourage linguistique) bien déterminé, là on parle d'un énoncé i.e. lorsque les conditions de vérification comportent l'environnement externe on passe d'une phrase simple à un énoncé unique.

Formellement, une phrase est conforme à un ensemble de règles grammaticales et syntaxiques bien définies et fixes, alors qu'un énoncé peut fournir plusieurs sens selon le contexte, la compréhension et l'interprétation donnés.[28]

a-Paraphrase sémantique ou linguistique

La paraphrase linguistique se base sur les correspondances syntaxiques et ou lexicales entre les phrases. On distingue deux types à savoir : les paraphrases lexicales et les paraphrases lexico-syntaxique.

- **Paraphrase syntaxique :** afin d'obtenir une paraphrase syntaxique, plusieurs transformations peuvent être utilisées dont la nominalisation, conversion d'un adjectif en syntagme nominal, l'épithétisation, transformation d'une proposition relative en adjectif épithète. Exemple : (« la zone de l'industrie » - « la zone industrielle »). Ce type de paraphrase se base donc sur une règle qui spécifie les conditions du passage d'une phrase à l'autre.
- **Paraphrase lexico-syntaxique :** Dans ce type de paraphrases, en plus des modifications opérées sur le niveau syntaxique, des modifications sur le niveau lexical sont effectuées.

La constitution de telles paraphrases peut être faite en appliquant une redistribution des arguments sur des actants différents ou bien en appliquant une double négation.
Exemple : « il a échoué à son examen » - « il n'a pas réussi son examen ».

b-Paraphrase non-linguistique

Contrairement aux paraphrases linguistiques, les paraphrases non linguistiques se basent sur des phrases comportant la même idée ou référant la même chose sans chercher des correspondances lexicales ou syntaxiques ce qui nécessite l'intervention de l'expert

- **Paraphrase pragmatique :** Deux paraphrases pragmatiques sont des phrases qui réfèrent à la même intention de telle sorte que les phrases sont interprétées de la même façon en se basant sur l'expérience et les connaissances.
  Exemple : « il fait chaud » - « je veux qu'on allume la climatisation ».
- **Paraphrase référentielle :** Dans ce type de paraphrases, il est nécessaire de connaitre les références de certains termes.
  Exemple : « la ville des roses » - « BLIDA ».

Dans notre cadre, le type de paraphrases à générer dépend du type de paraphrases que comporte le corpus d'entrainement. Cependant, nous estimons que les paraphrases à générer devront être de type paraphrases phrastique.

3-2 Les domaines d'application

De nos jours, la génération des paraphrases est une tâche très répandue dans le domaine du TAL. Elle est notamment utilisée dans différentes applications telles que :

- **La reformulation des requêtes dans la recherche d'informations (RI) :** la génération de paraphrases est utilisée fréquemment dans le domaine de RI. Le but étant de générer une variété de requêtes afin d'obtenir les résultats les plus pertinents. [26]
- **La traduction :** le système de traduction automatique peut être confronté à des difficultés lors de la traduction d'une phrase. L'utilisation d'une paraphrase comme phrase source peut constituer une solution permettant au système de trouver une bonne traduction.[26]
- **La similitude textuelle sémantique:** mesurer le degré de similitude dans le sens contextuel de deux phrases [27] (exemple: détection de plagiat)
- **Réponse aux questions (Question answering) :** ici les questions sont réécrites sous

forme de requêtes pour interroger une base de connaissances existante, celle-ci renvoie un ensemble de réponses, parmi elles, il faudra choisir celles qui sont correctes. Le système donnera à chaque réponse sélectionnée une valeur de similitude sémantique entre elle et la question. [27]

4- LES APPROCHES DE GENERATION DE PARAPHRASES

Plusieurs approches de génération existent. Nous pouvons les distinguer selon le corpus utilisé.

4-1 Approches fondées sur l'exploitation des ressources linguistiques

Dans ces approches, nous pouvons obtenir des paraphrases à l'aide de diverses ressources résultant des connaissances sémantiques. Ainsi certains mots peuvent en remplacer d'autres : la synonymie. On peut utiliser certaines ressources telle que Wordnet et/ou les encyclopédies afin d'en extraire les synonymes. Certaines langues, malheureusement, ne disposent pas de ces ressources tout comme pour la langue arabe.

Hormis ces ressources sémantiques, les dictionnaires généralistes et spécifiques (les dictionnaires de structures lexicales conceptuelles) semblent être utiles à l'extraction des équivalences sémantiques.

Il est essentiel de souligner que les normes de reconnaissance de paraphrases ne produisent pas des équivalences substituables dans tout contexte. Cette limite est forcément à l'origine des travaux qui nécessitent des connaissances depuis un corpus[26].

4-2 Approches de paraphrases fondées sur des corpus de données

La génération de paraphrase se repose ici ou bien sur le calcul de similarités en exploitant les propriétés lexicales, ou encore plus récemment sur l'utilisation des réseaux de neurones approfondis.

Dans les deux cas, les approches se fondent sur un corpus de données monolingues ou bilingues.

4-2-1 Approches basées sur les propriétés lexicales

Nous distinguons ici deux grandes sous catégories d'approches selon les langues des corpus utilisés.

a- corpus monolingue

La génération des paraphrases se base ici sur un seul corpus constitué d'unités de connaissances similaires se basant sur l'hypothèse distributionnelle[29]. Cette hypothèse suppose que chaque unité comporte au moins un segment de mots remplaçable par un autre segment de mots, ce qui

permet d'extraire des règles d'inférences représentables dans un arbre syntaxique comportant une relation binaire entre deux noms, il sera utilisé pour l'extraction de paraphrases [30]. Cependant, cette solution même très bien structurée, elle est très difficile et très couteuse à exploiter pour de grands corpus, Pasça et Dienes [31] ont proposé une approche mesurant la similarité avec la métrique n-grammes du contexte afin de diminuer le cout d'exploitation. Malgré l'utilisation de fonctionnalités de distribution plus informatives, cette approche peut générer plusieurs erreurs dans les modèles de paraphrase (règles d'inférence). D'autres travaux ont montré comment filtrer les inférences incorrectes [32]. Plusieurs autres travaux ont proposé des approches similaires [28], [33], [34].

D'autres techniques se sont basées sur l'exploitation de deux ou plusieurs corpus comparables associés en fonction d'une mesure de similarité textuelle. Ils se basent sur l'hypothèse que des unités linguistiques apparaissant plusieurs fois dans les textes similaires peuvent avoir la même signification. Dans cette optique, les auteurs dans [11] et [12] ont utilisé des corpus de journaux publiés dans la même période.

Les auteurs de [13] et [14] se sont appuyés, quant à eux, sur des algorithmes d'alignement combinant plusieurs heuristiques de similarités multi-séquences. Ils tentent d'identifier directement les correspondances dans deux corpus comparables monolingues.

Comme nous trouvons dans la littérature des approches fondées sur des paires d'énoncés équivalents alignées de façon supervisée ou semi supervisée. Ces approches sont utilisées dans [39] pour les traductions multiples et dans [40] pour les questions ayant la même réponse.

Les méthodes de génération de paraphrases sont passées par la suite, à l'utilisation des techniques de similarités statistiques en exploitant les propriétés lexicales comme Cosine Similarity, Jaccard Similarity, Resnik similarity, Lesk measure, Lch measure, Wu and Palmer Simliarity, etc. [27].

Parmi ces méthodes, nous citons l'approche de classification des unités de connaissance en couples. Elle se base sur le degré de proximités avec la valeur de la métrique TF-IDF (Term Frequency–Inverse Document Frequency) pour les textes du corpus[41]. La force de couplage est une estimation faite sur la base de prépositions et de conjonctions. Afin d'optimiser les performances de leur approche, les auteurs de cette approche ont utilisé des informations textuelles représentant une unité de connaissances sélectionnée compressée au moins deux fois en préservant sa signification (principe de compréhension d'images).

Kamal SARKAR [42] utilise un modèle de régression logistique multinomial formé avec une variété de caractéristiques, ces dernières sont essentiellement des similitudes lexicales et

sémantiques entre deux phrases d'une paire. Plusieurs techniques de similarité statistiques ont été utilisées. Cette solution a été testée sur quatre langues indiennes, et ça a donné un très bon résultat qui est supérieur à 0.9.

b- corpus multilingue

Ces approches se basent sur la présence des phrases disponibles dans au moins deux corpus de langues différentes. Elles utilisent une des langues comme pivot pour la génération de paraphrases en se basant sur l'hypothèse suivante : les unités de connaissances partageant la même traduction dans une autre langue peuvent être des paraphrases. Plusieurs travaux se basant sur ces corpus de traduction automatique statistiques existent dans la littérature [43], [44].

Al-Raisi, Lin et Bourai ont profité de cette technique pour construire un corpus de paraphrase en langue arabe [45], en se basant sur le corpus europarl-v7 [46] existant en anglais et en français comme pivot, traduisant le contenu des deux langues en arabe ça a induit à la création d'un corpus de paraphrases en langue arabe.

Ces approches permettent d'identifier un grand nombre de paraphrases. Cependant, elles sont limitées par les techniques d'alignements et aussi par la disponibilité des corpus parallèles appropriés dans les langues souhaitées.

4-2-2 Approches basées sur l'exploitation des ressources linguistiques

Afin d'améliorer les travaux existants, l'utilisation d'un corpus de données s'est avérée très primordiale. L'avancée technologique a permis la numérisation des ressources linguistiques. Ceci a induit l'apparition de nouvelles solutions de génération de paraphrases. Beaucoup d'entre elles sont liées aux nouvelles approches des réseaux de neurones.

L'apprentissage profond a été appliqué avec succès à diverses tâches de TAL ces dernières années. Plusieurs travaux appliquent efficacement les auto-encodeurs récursifs [47] et les réseaux de neurones convolutifs [48] pour la reconnaissance de la paraphrase. Cependant, la génération de paraphrases est une tâche plus difficile en raison à l'exigence de construire des alternatives sémantiquement similaires et grammaticalement précises à une phrase. Aucun travail jusqu'à présent, à notre connaissance, n'a tenté de résoudre ce problème pour la langue arabe en utilisant l'apprentissage profond, toutefois des travaux basés sur les RNN, le LSTM et leurs variantes [49]–[51] ont donné de bons résultats pour les autres langues surtout pour Anglais ou des corpus de données volumineux existent. Nous traiterons en détail les travaux les plus récents dans la section suivante.

4-3 La génération de paraphrase dans les systèmes ASAG

Nous citons dans cette section deux exemples de travaux basés sur l'exploitation des ressources linguistiques et qui n'ont pas introduit le Machine Learning dans leurs solutions.

- ✓ **Automatic Essay Grading System for Short Answers in English Language**

Les auteurs de ce travail [52] ont traité l'évaluation automatique des réponses courtes en se basant sur la génération des paraphrases. Ils ont proposé un système de classement (de notation) automatique des réponses courtes, ce dernier est basé sur un dictionnaire de synonymie sémantique. Les auteurs ont généré plusieurs réponses modèles à partir d'une réponse courte donnée, ils ont évalué avec une mesure de similarité la réponse des élèves par rapport à cet ensemble de réponses modèles générées.

Les auteurs ont proposé deux phases pour noter la réponse de l'élève : la première est une méthode alternative de génération de phrases, afin de générer la réponse alternative à une réponse modèle, en connectant la méthode au dictionnaire de synonymes. La seconde phase, est l'hybridation de trois algorithmes : Commons Words (COW), Longest Common Subsequence (LCS) et Semantic Distance (SD)(voir Annexe A) [52][53].

Evaluation. Cette solution a été testée sur une petite population (un examen de 40 questions posées à 3 élèves), elle a donné un taux de réussite de 82% par rapport à la notation de leur enseignant, comparé au système de ASAGS qui présente une corrélation de 60%.

- ✓ **Génération automatique de corpus de réponses modèles dans un système d'évaluation automatique des réponses courtes** [54]

Dans ce travail, les auteurs se sont inspirés du travail précèdent [52], ils ont appliqué le même principe pour la langue arabe en générant eux-mêmes un dictionnaire des synonymes. Ils ont construit, tout d'abord, un dictionnaire de synonymes arabophone à partir d'un corpus de données en utilisant l'hypothèse de distribution. Pour cela les auteurs ont choisi pour cela l'utilisation de deux techniques : l'espace sémantique et le word embedding pour la génération du dictionnaire. La combinaison des synonymes de mots a permis de varier, pour chaque réponse modèle, plusieurs réponses modèles de même sens.

Une fois le dictionnaire de synonymes construit, la génération de paraphrases est obtenue par combinaison de synonymes.

Le principe consiste à récupérer la réponse modèle (après avoir fait les prétraitements), les

synonymes de contexte pour chaque mot dans le dictionnaire de synonymes, et d'appliquer plusieurs combinaisons entre ces synonymes pour avoir plusieurs paraphrases.

Finalement, les auteurs ont calculé la similarité entre la réponse de l'élève et les réponses modèles (les paraphrases générées), en appliquant deux modèles de similarité à savoir, le modèle somme-vecteurs avec et sans pondération [55], et Le modèle calcul-matriciel avec et sans pondération [56]. Quant au score final, il a été calculé en appliquant l'algorithme K-means [57] (avec k = 11), et ce, pour convertir les valeurs de similarité comprises entre 0 et 1 en des valeurs comprises entre 0 et 5(les scores).

Nous avons résumé l'ensemble de ces étapes dans le schéma suivant (Figure II.1):

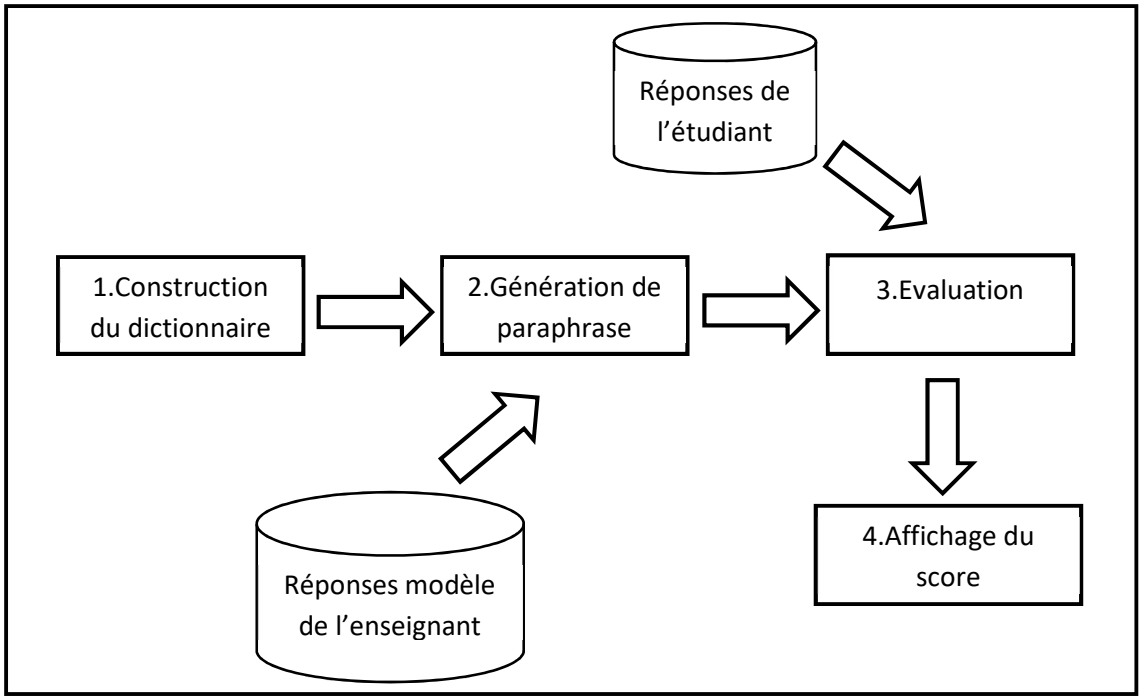


Figure II-1 Processus d'évaluation automatique des réponses à base de dictionnaire sémantique

Evaluation. Les auteurs de ce travail ont testé leur solution sur corpus comportant 48 questions, avec 1233 réponses au total. Le taux de corrélation est de 67,87% par rapport à la notation manuelle des enseignants.

5- REVUE DE LA LITTERATURE-GENERATION DE PARAPHRASES

La génération automatique des paraphrases est une tâche très importante du TAL, d'ailleurs elle est de plus en plus utilisée pour améliorer les performances de plusieurs applications en TAL tel que la traduction et la reformulation des requêtes. Dans ce cadre, nous avons recensé plusieurs travaux de recherche dans la littérature, toutefois nous nous sommes intéressés qu'aux articles les plus récents qui ont traité plus l'aspect supervisé de la paraphrase et qui seront illustrés dans cette section.

Toutes les approches que nous présentons ici sont supervisées et par conséquent nous commençons par faire une présentation des jeux de test(datasets) ainsi que les métriques les plus utilisées.

Pour chacun des travaux étudiés, nous présentons l'approche, les datasets utilisés pour l'entrainement, l'expérimentation menée ainsi que l'évaluation des résultats obtenus. Nous dresserons une synthèse comparative de l'ensemble des travaux présentés en fin de cette section.

5-1 Datasets

Plusieurs corpus de données ont été développés ces dernières années pour être utilisés dans les applications du TAL et autre, ils ont été approuvés par la communauté scientifique. Nous citons ici, les corpus les plus usuels dans les langues arabe et anglaise.

5-1-1 PPDB (Pesticides Properties DataBase)

PPDB [71] est un dataset de paraphrases, conçu dans le but d'offrir un large corpus de données pour diverses applications du traitement de la langue. PPDB* est disponible en plusieurs tailles et en 16 langues différentes y compris l'arabe. Ce corpus comporte des paraphrases de type lexical, syntaxique et phrastique dont la plupart sont courtes.

5-1-2 WikiAnswers

WikiAnswers [72] est un dataset de 18 millions de paires de questions extraites à partir des questions posées par les internautes du site WikiAnswers. Les questions jugées identiques par ces internautes ont été considérées comme des paraphrases.

* Disponible sur http://paraphrase.org/#/download

5-1-3 MSCOCO

MSCOCO [73] est un dataset qui contient un ensemble d'annotations données sur plus de 120 000 images. Chaque image a été annoté par 5 annotateurs différents, où les 5 annotations d'une même image sont considérées comme paraphrases (des phrases ayant la même signification).

5-1-4 Quora

Quora [74] est un dataset qui contient 400 000 paires de questions qui portent sur des sujets différents où chaque deux phrases d'une même paire sont des paraphrases.

5-1-5 PARANMT50M

PARANMT50M [75] est un corpus de paraphrases contenant plus de 50 millions de paires de paraphrases. Ce corpus a été construit en appliquant la traduction automatique sur les phrases tchèques d'un corpus parallèle bilingue tchèque-anglais et en rajoutant les phrases références (anglaises) afin de former des couples de paraphrases.

5-1-6 EuroParl

EuroParl [46] est un corpus bilingue extrait à partir des travaux du parlement européen dont le but est la réutilisation dans les taches de traduction automatique.

5-1-7 A Monolingual Parallel Corpus of Arabic

Ce corpus monolingue parallèle [45] a été généré automatiquement en traduisant le corpus parallèle bilingue EuroParl-v7 Anglais-Français en arabe via l'API de google traduction. Ce dataset a été évalué par deux experts de la langue arabe et peut être utilisé dans le cadre du TAL, notamment dans l'entrainement des modèles de séquences qui traitent la tache de génération des paraphrases.

5-2 Les métriques usuelles

L'évaluation automatique consiste à calculer le taux de similarité entre la paraphrase générée et celle du dataset, ainsi que le nombre de mots clés apparus dans la paraphrase. Quant à l'évaluation humaine permet, d'une part, de vérifier les paraphrases grammaticalement et sémantiquement, d'autre part, de dire si le modèle peut générer des différentes paraphrases en se basant sur des ensembles de mots clés différents.

Dans le but de mesurer la qualité d'un système de génération de paraphrases, plusieurs métriques peuvent être utilisées dont PINC [76] et PEM [77].

Cependant, ces métriques présentent certaines limites, ce qui amène les chercheurs à utiliser des mesures de traduction automatique tel que les METEOR, TER, BLEU, WER et GLEU, et ce, pour mesurer la qualité des paraphrases. Nous présentons ces métriques dans ce qui suit.

5-2-1 METEOR

METEOR [61] est une mesure utilisée principalement pour l'évaluation de la traduction automatique qui assure une grande corrélation avec le jugement humain. Cette métrique permet la résolution de plusieurs faiblesses apparues dans la métrique BLEU [60].

Le principe de base de cette métrique est de trouver un alignement entre les mots de la phrase candidate et celle de référence. La correspondance entre les deux phrases est trouvée selon trois modules:

- Le module « exact »: juge deux mots équivalents s'ils sont exactement les mêmes.
- Le module « porter stem »: deux mots sont équivalents s'ils sont exactement les mêmes après avoir appliqué le stemmer de porter.
- Le module « WN synonymy »: deux mots sont jugés équivalents s'ils appartiennent au même synset dans le WordNet (s'ils sont synonymes).

Après avoir trouvé l'alignement qui préserve le plus l'ordre des mots présent dans les deux phrases, il faut calculer la moyenne harmonique paramétrée comme suit :

$$F_{\text{mean}} = \frac{P.R}{\alpha.P+(1-\alpha).R}$$

Où la précision des uni grammes P = m/t et le rappel des uni grammes R = m/r, tel que (m) représente le nombre d'uni-grammes mappés entre les deux phrases, le (t) représente le nombre d'uni grammes dans la phrase de référence et le (r) représente le nombre d'uni grammes dans la phrase candidate.

Afin de prendre en compte le degré de correspondance d'ordre des mots présents dans l'alignement par rapport à ceux présents dans les deux phrases (référence et candidate), on divise cet ensemble de mots (de l'alignement) en un ensemble de morceaux (le plus petit ensemble possible), tel que chaque morceau contient des uni grammes adjacents et qui sont

dans le même ordre dans la phrase référence et candidate. Une fois ce travail a été fait, une pénalité est calculée comme suit :

$$Pen = \gamma . frag^{\beta}$$

Tel que frag =ch/m, (ch) représente le nombre de morceaux, β détermine la relation fonctionnelle entre fragmentation et la pénalité et γ représente la valeur maximale de pénalité comprise entre 0 et 1.

Les valeurs des paramètres ont été fixé à : $\alpha = 0.9$, $\beta = 3.0$ et $\gamma = 0.5$, et ce, en se basant sur les différentes expérimentations effectuées précédemment.

Le score final de METEOR est calculé comme suit :

$$score = (1 - Pen).F_{mean}$$

5-2-2 TER

TER (Translation Edit Rate) [62] est une métrique utilisée pour l'évaluation de la traduction automatique. Elle mesure le taux minimal de modifications nécessaires sur la phrase candidate afin d'obtenir une phrase similaire à l'une des phrases références i.e. plus les deux phrases se ressemblent moins sera le nombre de modifications effectués.

Les modifications englobent :

- L'insertion : un mot qui apparait dans la phrase candidate et n'apparait pas dans la phrase référence.
- La suppression : désigne l'ensemble des mots qui apparaissent dans la phrase référence et n'apparaissent pas dans la phrase candidate.
- La substitution : c'est le fait de remplacer un mot de la phrase référence par un autre dans la phrase candidate.
- Le décalage : désigne l'inversion d'ordre des séquences de mots entre les deux phrases.
- La ponctuation ainsi que la capitalisation sont aussi prises en considération.

Le score de TER est calculé comme suit :

$$\text{TER} = \frac{Nombre\ de\ modifications}{Le\ nombre\ moyen\ des\ tailles\ des\ phras \quad références}$$

5-2-3 BLEU

BLEU (BiLingual Evaluation Understudy) [60] est l'une des métriques utilisées dans le cadre d'évaluation automatique de la traduction. Son principe de base consiste à compter le nombre des n-grammes (uni-gramme, bi-grammes, trigrammes et quadrigrammes) de la phrase candidate présents dans les phrases de références. Ce nombre est exprimé par le calcul de la précision standard P = n /c tel que :

(n): Nombre de n-grammes de la phrase candidate présents dans les références.

(c): Nombre de n-grammes de la phrase candidate.

La comparaison n-grammes à n-grammes peut induire à une précision élevée dans le cas où un n-grammes présent dans une phrase référence se répète plusieurs fois dans la phrase candidate. Ceci représente une limite lors du calcul de la précision.

Exemple :

La phrase candidate : Livre livre livre livre.

La phrase référence : Le livre est sur la table.

Dans cet exemple la précision standard P = 4/4, ceci est dû à la répétition du mot « livre ».
Pour remédier à ce problème, le nombre de n-grammes est réduit au nombre maximal de n-grammes présents dans la référence. La nouvelle formule de précision (appelée précision modifiée) est calculée en faisant la somme des fractions f = n/c tel que :

- (n) est égal au nombre d'occurrence du n-gramme dans la phrase candidate, si le nombre d'occurrences du n-gramme dans la phrase candidate ≤ le nombre d'occurrences du n-gramme dans la phrase référence.
- Sinon, (n) est égal au nombre d'occurrence du n-gramme dans la phrase référence.

En appliquant cette formule sur l'exemple précédent, on obtient une valeur de précision P=1/4, ce qui est raisonnable.

Afin de calculer la précision dans plusieurs blocs de texte, on découpe les blocs de texte en phrases, celles-ci représentent l'unité de base du calcul. On procède par le calcul des n-grammes correspondant entre les phrases (candidates et références), ensuite on calcul la somme des n-grammes réduits cette dernière est divisée par le nombre total des n-grammes des phrases candidates.

$$Pn = \frac{\sum_{C\in\{Candidates\}} \sum_{n-gra \quad \in C} Count_{clip}\,(n-gram)}{\sum_{C\in\{Candidates\}} \sum_{n-gra \quad \in C} Count\,(n-gram)}$$

Les phrases candidates qui ont une taille plus longue que les phrases références sont pénalisées lors du calcul de la précision modifiée, contrairement aux phrases courtes, qui peuvent avoir une précision élevée, alors qu'elles ne représentent pas toute la signification de la phrase référence. Pour cela, un facteur de pénalité BP (Brevity Penalty factor) a été appliqué afin de pénaliser que les phrases courtes. Ce facteur est calculé comme suit :

$$\text{BP} = \begin{cases} 1 & si\ c\ > r \\ e^{(1\ -r\ /\ c)} & si\ c \leq\ r \end{cases}$$

Où (r) représente le nombre de n-grammes dans la phrase référence.

Le score de la métrique BLEU est calculé selon la formule ci-dessous :

$$BLEU\ =\ BP.exp(\sum_{n=1}^{N} w_n\ log\ p_n)$$

Au final, la fonction log est appliquée pour avoir un résultat plus apparent :

$$log\ BLEU\ =\ min\,(1\ -\ \frac{r}{c}\ ,0)\ +\sum_{n=1}^{N} w_n\ log\ p_n$$

w_n: Représente les poids.

5-2-4 GLEU

La métrique GLEU (Google-BLEU) est une variante de la métrique BLEU, utilisée spécialement pour mesurer taux de corrections d'erreurs grammaticales des n-grammes générés avec l'ensemble des phrases références. Il a été prouvé que les résultats générés par cette métrique sont proches de ceux relatifs aux humains [78], [79].

5-2-5 WER

La métrique WER (Word Rate Error) est utilisée pour mesurer la performance des systèmes qui traitent la langue naturelle, tel que les systèmes de traduction automatique [80].

Cette métrique, dérivée de la distance de Levenshtein, travaille sur le niveau des mots, elle est très utilisée pour faire des comparaisons entre systèmes. Elle compare entre deux phrases, référence et hypothèse, en prenant en considérations un certain nombre de critères :

- Le nombre de mots insérés I dans la phrase hypothèse, et qui n'appartiennent pas à la phrase référence (les mots additionnés).
- Le nombre de mots substitués S qui représente le nombre de mots de la phrase hypothèse, qui ont été remplacés par d'autres mots dans la phrase source.
- Le nombre de mots supprimés D.

Une petite valeur de WER indique que le système est de meilleure qualité. Sa valeur est calculée en appliquant la formule suivante :

$$WER = \frac{S + D + I}{N}$$

Où N représente le nombre de mots de la phrase référence.

5-3 Les travaux connexes

Plusieurs travaux traitant la génération de paraphrases ont été recensés dans la littérature, nous nous intéressons dans cette section aux travaux basés sur les approches de Deep Learning.

5-3-1 Neural Paraphrase Generation with Stacked Residual LSTM Networks (Residual LSTM) [50]

Dans le cadre d'un travail de génération automatique des paraphrases via les réseaux de neurones, Aaditya Prakash et al [50] ont utilisé une technique qui consiste à empiler plusieurs couches LSTMs auxquelles ils ajoutent des connexions résiduelles. Le but est d'obtenir de meilleures performances à partir de réseaux neuronaux très profonds.

Des études ont montré que plus un réseau de neurones est profond, meilleure sera sa performance i.e. la performance du réseau de neurones varie d'une façon proportionnelle en fonction du nombre de couches. Cependant la forte augmentation de nombre de couches entraine la saturation voire même la dégradation des performances du réseau, elle impose par conséquence, la disponibilité d'ensembles de données de tailles importantes. Les connexions résiduelles peuvent aider à surmonter le problème de dégradation dû au faible taux de convergence des erreurs d'entrainement. Ce problème est principalement causé par l'empilement de plusieurs couches de neurones. Cette solution inspirée de ResNet [58] consiste à additionner explicitement des résidus x à la fonction en cours d'apprentissage tout en évitant le sur-apprentissage.

Les auteurs ont opté pour un empilement vertical des LSTM où seule la sortie de la couche

précédente de LSTM est envoyée à l'entrée, comme le montre la Figure II.2. Ils ont utilisé quatre couches LSTM empilées dont les connexions résiduelles ont été ajoutées à la couche 2.

Une connexion résiduelle est ajoutée toutes les n couches. Dans le cadre de ce travail, la valeur de n est égale à 2, ce qui est généralement conseillé car la complexité des calculs devient importante dès que n dépasse 3.

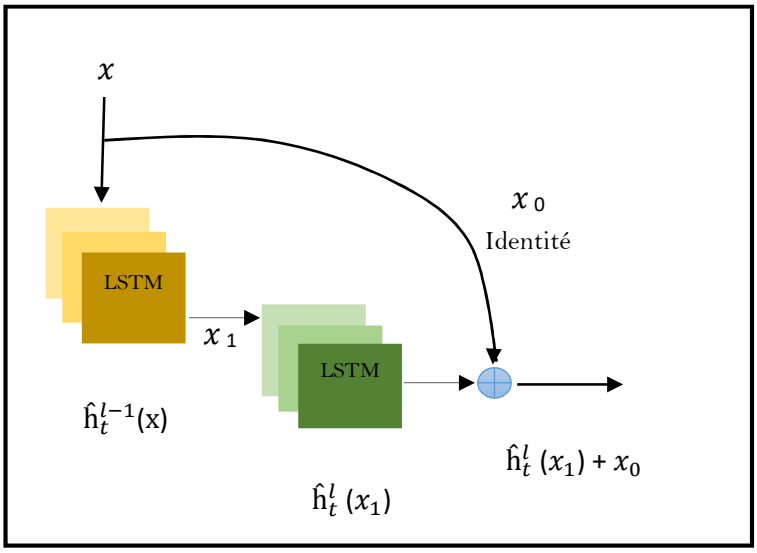


Figure II-2 Unité d'un LSTM résiduel empilé[50]

Ensembles de données d'entrainement (datasets)

Afin de mieux évaluer le modèle, trois datasets différents en termes de caractéristiques ont été utilisé dont, PPDB, MSCOCO et WikiAnswers.

Expérimentation.

L'échantillon d'expérimentation a été extrait des trois datasets cités précédemment (PPDB, MSCOCO et WikiAnswers). Ce dernier est devisé en deux parties distinctes, l'une dédiée à l'apprentissage et l'autre au test.

Les modèles utilisés lors de l'expérimentation ont été entrainés en utilisant un algorithme de descente de gradient stochastique et la fonction de perte perplexity [59]. Une représentation One-hot* a été appliquée sur les mots du vocabulaire. 512 unités LSTM par couche ont été utilisées

* One Hot ou l'encodage à chaud est une représentation de mots sous forme de vecteurs binaires. Chaque valeur entière (correspond à un mot donné) est représentée sous la forme d'un vecteur binaire dont toutes les valeurs sont nulles, à l'exception de l'indice de l'entier, qui est marqué de un (1)

pour chacun des modèles.

Evaluation

A.Prakash et al., ont utilisé les métriques Emb Greedy qui utilise word embedding pour comparer les phrases, BLEU [60], METEOR [61] et TER [62] dans le but d'évaluer et de comparer les modèles.

Les résultats d'expérimentation ont montré que :

- Pour la génération de plusieurs paraphrases à la fois (Beam size = 10), le modèle LSTM résiduel a donné les meilleurs résultats sur WikiAnswers et MSCOCO par rapport à toutes les métriques.
- En évaluant les modèles sur PPDB, METEOR est la seule métrique où les résultats du LSTM résiduel ont été dépassés. Ce fait est dû à la taille réduite de phrases contenues dans PPDB.

5-3-2 Dictionary-Guided Editing Networks for Paraphrase Generation (DGEN) [51]

En 2018, Shaohan Huang et al [51] ont proposé une approche basée sur un dictionnaire permettant la génération automatique des paraphrases. Le principe général de cette approche est de sélectionner quelques mots ou passages et de les remplacer par d'autres ayant le même sens (en se basant sur un dictionnaire. Cette solution est inspirée de la façon dont les humains écrivent des paraphrases.

La première étape de l'approche consiste à sélectionner des mots et passages de la phrase source afin de récupérer leurs paraphrases. Elle se base sur un dictionnaire phrastique et lexical.

$$\varepsilon = \{(0i\ , P_i)\}_{i=1}^{M}$$

Où ε est l'ensemble des pairs de mots/passages (Oi et Pi) ayant le même sens, obtenus à partir du dictionnaire.

Une représentation vectorielle de l'ensemble des paires est nécessaire afin de les manipuler lors de la prochaine étape. Cette représentation qu'on notera ε' est faite via un encodeur de dictionnaire. Lors de la dernière étape, un modèle de réseau d'édition guidé par dictionnaire avec un mécanisme d'attention permet de guider la décision de supprimer ou d'ajouter des mots à chaque étape du décodeur.

L'architecture du modèle DGEN est illustrée dans la figure II.3.

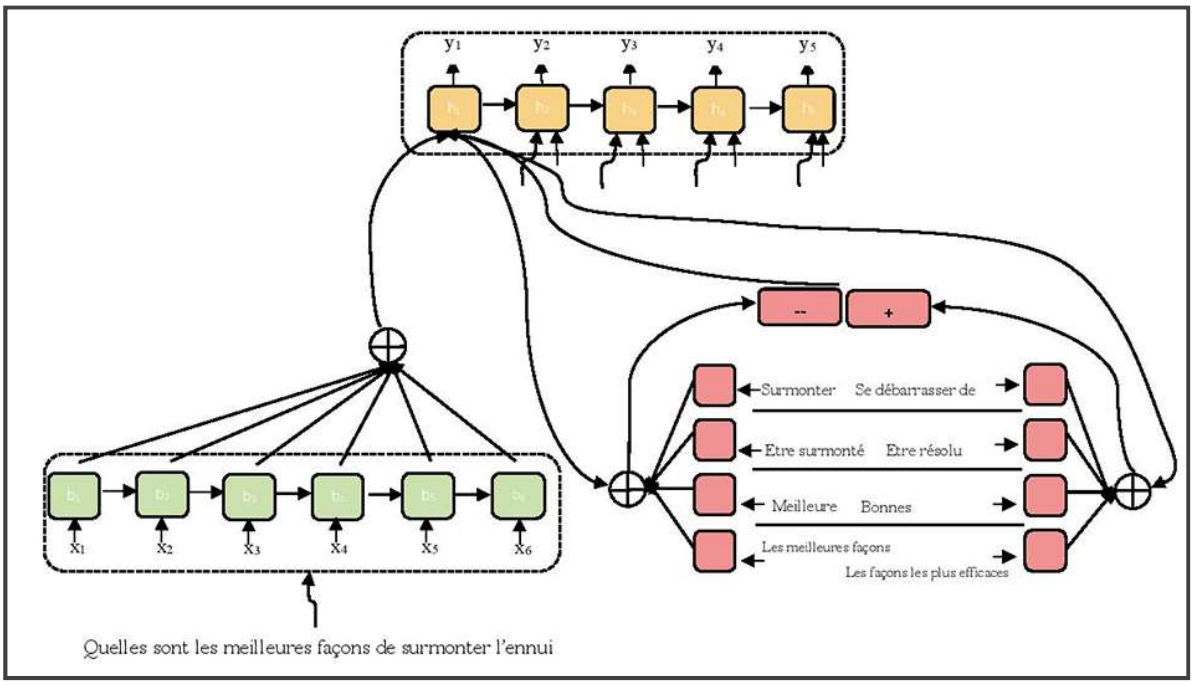


Figure II-3 Architecture du réseau d'édition guidé par dictionnaire[51]

Expérimentation

- L'approche de représentation des mots est l'approche one hot.
- Les métriques d'évaluations BLEU et METEOR ont été appliquées en utilisant les datasets MSCOCO et QUORA afin d'évaluer le modèle.
- Le dictionnaire utilisé par cette approche a été construit sur la base du dataset PPDB (taille L).
- Une étape de tokenisation a été appliquée afin de conserver les mots qui apparaissent plus de dix fois en utilisant NLTK [63].
- Un prétraitement a été fait sur les annotations de MSCOCO afin de réduire leurs tailles aux 15 premiers mots, et les phrases du dataset Quora dont la taille dépasse 30 mots ont été supprimées.
- Les dimensions du vecteur caché, word embedding ainsi que le vecteur d'attention ont été fixés respectivement à 512, 300 et 512.

Evaluation

La méthode de Shaohan Huang et al a été comparée aux modèles Seq2Seq [64], Residual LSTM [50], VAR-SVG [65] et VAE-SVG-eq [65].

Les résultats de l'évaluation ont montré que Dictionary-Guided Editing Networks a dépassé toutes les autres méthodes sur le dataset MSCOCO pour un beam size égal à 1 et 10 et sur le dataset Quora avec un beam size de 1. Par contre, pour un beam size égal à 10, cette méthode

a été dépassée seulement par VAE-SVG-eq sur le dataset Quora.

5-3-3 User-Oriented Paraphrase Generation with Keywords Controlled Network (KCN) [49]

DAOJIAN ZENG et al [49] ont proposé une nouvelle approche du Deep Learning où la génération de paraphrases est contrôlée par des mots clés. Le modèle est basé sur un framework « sequence-to-sequence ». La phrase source et l'ensemble des mots clés sont représentés sous forme vectorielle, et ce, en utilisant deux encodeurs (l'un pour la phrase source et l'autre est dédié aux mots clés). La structure est une combinaison des RNNs bidirectionnels et de GRU avec des poids différents.

Les deux vecteurs de contexte obtenus de l'encodage sont fusionnés pour qu'ils puissent être utilisés comme entrée du décodeur. La phase de décodage avec mécanisme d'attention (chapitre I section 6) consiste à générer la phrase cible (paraphrase) en copiant un ensemble des mots clés ou en générant des mots à partir du dictionnaire, et en prenant en considération la phrase source.

Pendant la phase d'entrainement, l'extraction de mots clés $< k^s, k^t >$ à partir des phrases sources et cibles $< s^s, s^t >$ est faite en combinant deux modèles PositionRank [66] et RandomSample [49]. La Figure II.4 représente l'architecture du modèle KCN.

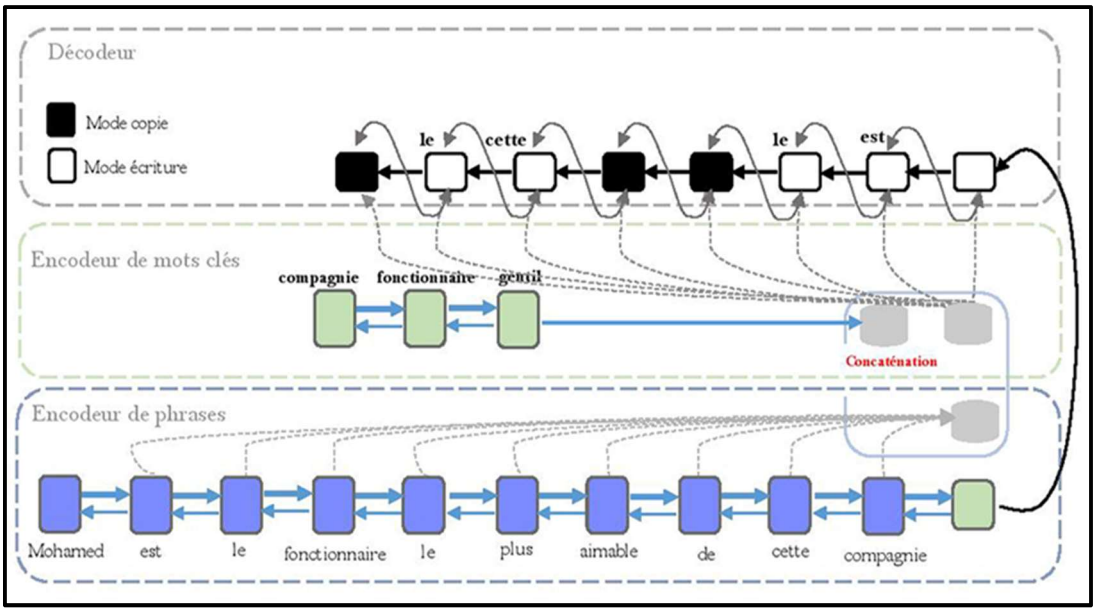


Figure II-4 Présentation du réseau contrôlé par mots-clés (KCN)[49]

Expérimentation

L'évaluation du modèle a été faite par rapport les métriques BLEUE, TER et METEOR sur les datasets PARANMT50M, MSCOCO et QUORA. Une autre évaluation de type humain a été faite afin d'assurer une vérification plus précise sur les paraphrases obtenues.

Evaluation

Les résultats ont montré que le KCN dépasse tous les modèles dont le modèle DGEN et Residual LSTM selon les métriques BLUE, METEOR et TER sur le dataset MSCOCO.

5-3-4 Paraphrasing Arabic Metaphor with Neural Machine Translation (PAMNMT)[67]

Dans un travail de génération de paraphrases consacré à la langue arabe, Manar Alkhatib et Khaled Shaalan [67] ont testé une approche qui consiste à générer des paraphrases en introduisant la traduction automatique. Cette approche se focalise beaucoup plus sur les métaphores, l'idée est de traduire cette dernière de l'arabe vers une autre langue « langue pivot », et ce, en utilisant un corpus bilingue. La langue pivot choisie dans ce travail est l'anglais. Une fois la phrase anglaise obtenue, elle sera traduite en arabe en se basant sur les distributions de probabilités. La difficulté réside dans la traduction automatique, malgré les résultats avancés obtenus dans ce domaine, la langue arabe reste une langue complexe qui manque énormément de ressources (corpus…).

Alkhatib et Shaalan ont utilisé l'une des méthodes NMT (Neural Machine Translation) [68] qui est basée sur deux réseaux de neurones récurrents (Encodeur-Décodeur). L'encodeur permet de générer un vecteur de représentation (pour chaque mot en entrée), le décodeur quant à lui il génère les mots dans la langue cible en prenant ce qui a été généré par l'encodeur.

L'encodeur est un RNN bidirectionnel qui prend une phrase source (séquence de mots) et génère un ensemble de vecteurs de contexte. Le décodeur est un RNN conditionnel qui génère une distribution de la probabilité sur la traduction à partir d'une phrase source.

La méthode Pivot [43] utilisée dans ce travail présente une solution lors de l'absence d'un chemin directe entre la langue source et la langue cible.

Evaluation

Le dataset d'entrainement est un corpus bilingue comportant 90 000 métaphores arabes avec leurs traductions anglaises extraites à partir des livres de rhétorique arabe.

L'évaluation a été effectuée en appliquant la métrique METEOR afin d'évaluer la qualité de la traduction arabe-anglais et anglais-arabe. Selon leur comparaison, les résultats obtenus étaient meilleurs que ceux de Google Traduction.

Une autre comparaison a été effectuée entre les paraphrases générées par ce modèle et celles générées manuellement par des experts. L'évaluation est faite avec la mesure Cosinus, elle a montré une forte corrélation avec un pourcentage très élevé, il a atteint 92.1% pour la traduction anglais- arabe et 89.9% pour la traduction arabe-anglais.

Nous constatons que l'évaluation de cette méthode a été faite sur la traduction anglais-arabe et arabe-anglais et non pas sur le processus de paraphrase lui-même.

5-3-5 Neural Symbolic Arabic Paraphrasing with Automatic Evaluation (NSAPAE) [69]

En 2018, F.Al-Raisi, A. Bourai et W.Lin [69] ont proposé deux approches de génération des paraphrases en langue arabe. La première est basée sur la traduction d'un corpus parallèle bilingue (Anglais-Français).

L'idée est de traduire chacune des paires en langue arabe (anglais -> arabe, et français -> arabe) en utilisant Google translate API, ce qui permet de générer un dataset de paraphrases arabes.

La deuxième approche est basée sur les réseaux de neurones LSTM bidirectionnel. Etant donné le manque de datastets arabe, les auteurs ont alimenté le modèle avec le corpus monolingue parallèle généré dans la première approche lors la phase d'entrainement. La figure ci-dessous montre le processus de génération du corpus parallèle arabe.

La figure II.5 montre le processus de génération du corpus parallèle arabe.

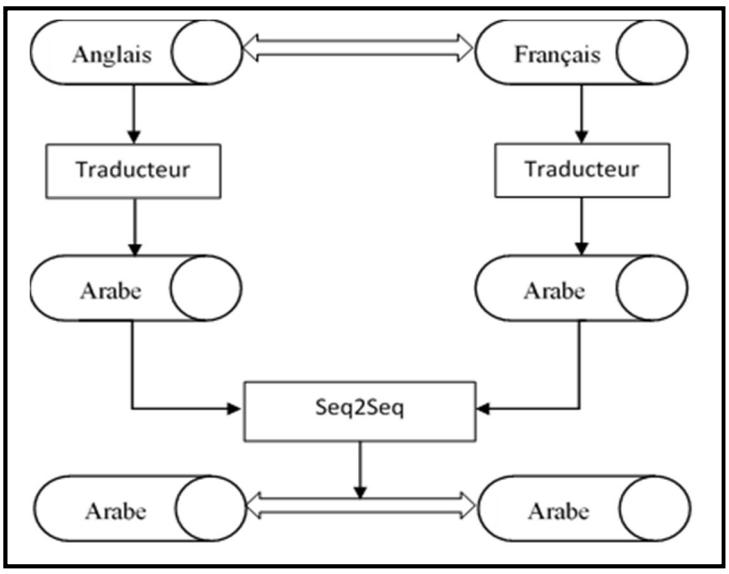


Figure II-5 Aperçu du processus de génération de deux corpus arabes parallèles à partir de différentes langages sources [69]

Evaluation

Le dataset europarl-v7 français-anglais [46] contenant deux millions de pairs de phrases a été exploité dans le but de générer le corpus monolingue.

Le dataset généré a été évalué grammaticalement et sémantiquement par deux experts en langue arabe, les résultats s'avèrent être encourageants. 62% des paraphrases ont été jugées correctes (avec un score 5/5), et ce, pour les deux critères.

Par contre, les résultats de la deuxième approche étaient loin d'être bons, malgré que le modèle a appris quelques caractéristiques des phrases.

L'évaluation humaine étant longue, a été renforcée par une évaluation automatique en employant trois métriques à savoir la métrique Cosinus, la variation de surface et la combinaison des critères de forme et de signification. Cependant aucun résultat n'a été publié.

5-3-6 Deep Generative Framework for Paraphrase Generation (VAE-AVG) [70]

En 2017, A.Gupta et al [70] ont proposé un Framework génératif profond pour la génération des paraphrases. Ce dernier se base principalement sur le modèle VAE (variational autoencoder), dont l'encodeur et le décodeur ont la structure d'un modèle sequence-to-sequence LSTM, ce modèle permet de générer plusieurs paraphrases sémantiquement équivalentes à la phrase d'origine. Afin que les paraphrases générées couvrent toute la sémantique de la phrase

source, cette dernière conditionne les deux parties du VAE à savoir le codeur et le décodeur.

Le dataset d'entrainement est constitué de N paires de paraphrases, tel que $S^{(o)}$ représente la phrase d'origine et $S^{(p)}$ représente la paraphrase. Chacune de ces dernières est composée d'un ensemble de mots. Les représentations vectorielles de la phrase source et la phrase cible sont obtenues via les réseaux de neurones LSTM et sont notées $x^{(o)}$et $x^{(p)}$ respectivement.

L'architecture globale du modèle proposé (où les réseaux de neurones LSTM ont été négligés pour des raisons de simplification) est illustrée dans la figure II.6.

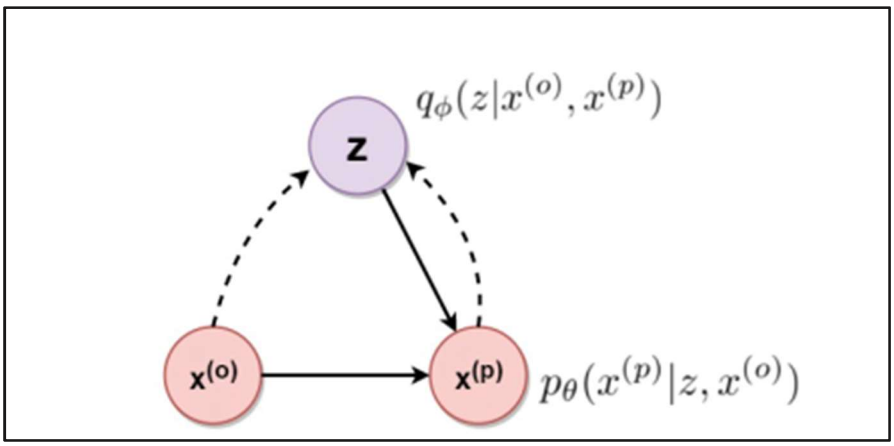


Figure II-6 Une macro-vue du modèle VAE LSTM [70]

Cette figure montre que la génération de $x^{(p)}$ nécessite le code latent z ainsi que $x^{(o)}$, le modèle du décodeur $p\theta(x^{(p)}|z, x^{(o)})$ est conditionné par la représentation vectorielle de la phrase source $x^{(o)}$.

La figure II.7 montre la représentation détaillée du modèle (avec LSTM), et ce, en décrivant la structure du codeur et du décodeur.

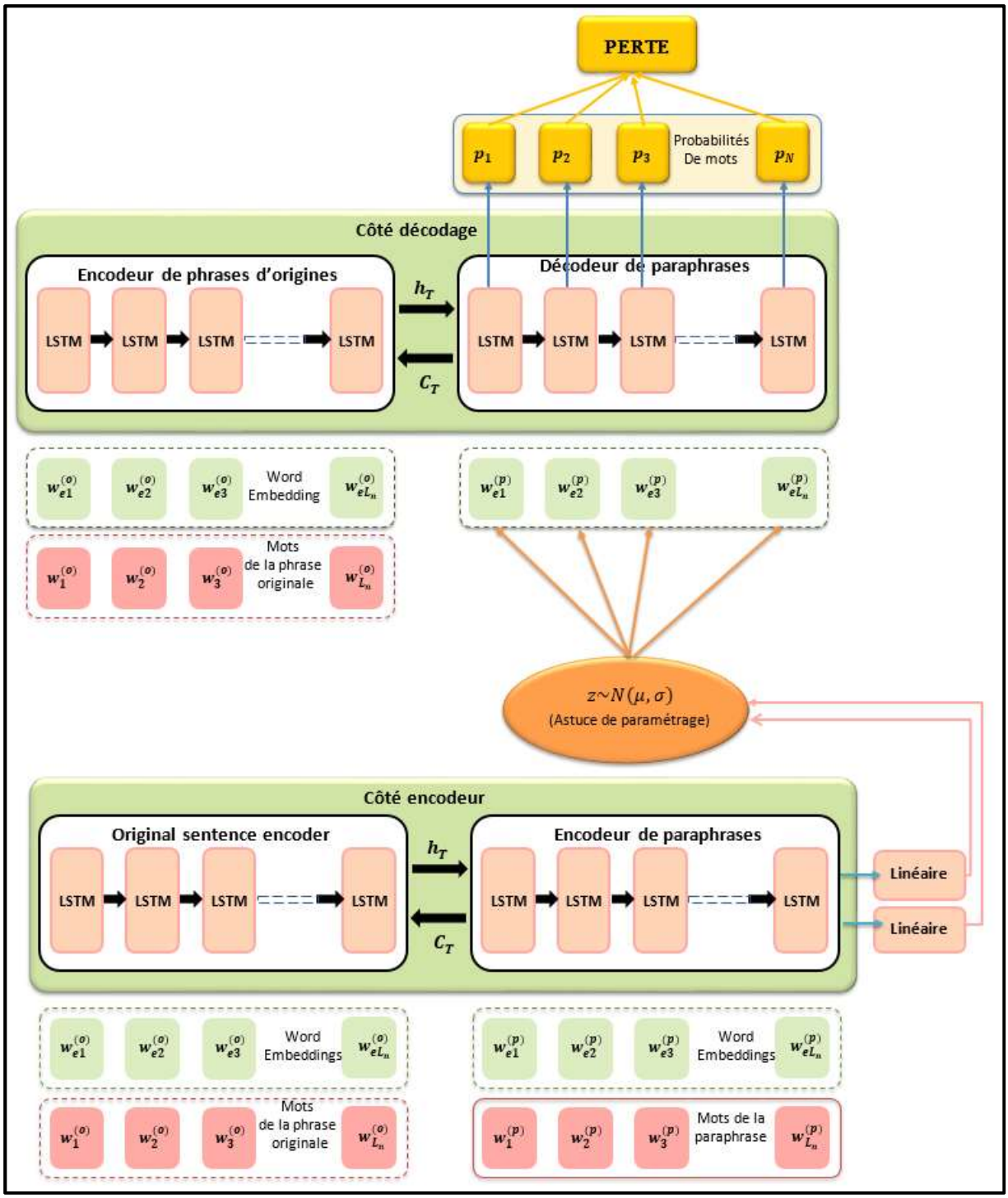


Figure II-7 Le schéma de l'architecture VAE-LSTM pour la génération de paraphrase[70]

L'obtention de la paraphrase à partir d'une phrase source passe par les étapes suivantes :

- La phrase source $s^{(o)}$ passe par un premier codeur LSTM, et ce, pour obtenir sa représentation vectorielle $x^{(o)}$.
- Le résultat de l'étape précédente ainsi que la phrase $s^{(p)}$ sont utilisés pour alimenter un autre codeur LSTM, le but est de trouver la représentation vectorielle $x^{(p)}$ de la

paraphrase. Cette dernière est utilisée dans un réseau de neurones afin d'obtenir la moyenne et la variance du codeur.

- Dans la phase de décodage, $x^{(o)}$ obtenu par un troisième codeur et le code latent z obtenu lors de la phase du codage sont utilisés comme entrées du décodeur LSTM, cette phase permet de générer la paraphrase.

Expérimentation

La taille du vecteur représentant le word embedding de chaque mot a été fixée à 300. La dimension du codeur et du décodeur est de 600, et celle de l'espace latent est de 1100. Le nombre de couches de l'encodeur et du décodeur sont 1 et 2, respectivement. Les datasets utilisés lors de l'entraînement du modèle sont MSCOCO et Quora. Quant aux métriques d'évaluation, ils ont utilisé BLEU, METEOR et TER.

Evaluation

Deux types d'évaluation ont été effectués :

- **Evaluation qualitative :** Consiste principalement à évaluer la qualité des paraphrases par les humains (experts), ces derniers donnent un score de 1-5 qui représente la pertinence de la paraphrase générée.
- **Evaluation quantitative :** Pour ce type d'évaluation automatique, les métriques BLEU, METEOR et TER ont été utilisées. Le modèle proposé a montré des performances qui dépassent le modèle VAE-S (variante supervisée), et ce, pour les deux datasets MSCOCO et Quora.

5-4 Synthèse des travaux

Nous avons essayé en premier temps de récupérer les résultats d'évaluation obtenus dans les différents travaux connexes cités dans ce chapitre, selon les métriques usuelles résumées précédemment.

Nous illustrons leurs résultats dans le tableau (II.1) qui englobe les performances des modèles sur le dataset MSCOCO et QUORA.

Tableau II-1 Evaluation des résultats des différentes méthodes sur les dataset MSCOCO et QUORA

	MSCOCO			QUORA	
Modèle	BLEU ↑	METEOR ↑	TER ↓	BLEU ↑	METEOR ↑
KCN [49]	**48.5**	**39.1**	**32.7**	N/A	N/A
Residual LSTM [50]	37.0	27.0	51.6	26.3	26.2
DGEN [51]	42.6	31.3	N/A	**27.6**	**29.9**
VAE-SVG [70]	41.7	31.0	40.8	26.2	25.7
Seq2Seq [20]	33.4	25.2	53.8	25.9	25.8
PAMNMT [67]	N/A	N/A	N/A	N/A	N/A
NSAPAE [69]	N/A	N/A	N/A	N/A	N/A

Nous constatons suite à ce tableau que le modèle KCN présente les meilleurs résultats d'évaluation pour les trois métriques citées pour le dataset MSCOCO.

Pour le dataset QUORA, en excluant KCN qui n'a pas utilisé ce dataset, DGEN, classé en deuxième position dans MSCOCO, a donné les meilleurs résultats par rapport aux autres approches.

Toutefois, ceci nous est insuffisant pour choisir un des modèles comme référence pour notre travail. Pour approfondir notre étude, nous avons effectué une étude comparative multicritères sur les différents modèles étudiés précédemment.

Le tableau de synthèse (II.2) illustre les résultats de notre comparaison après avoir retenu les critères suivants :

- Langue de paraphrases
- Année de publication
- Type de réseau de neurones
- Utilisation du mécanisme d'attention ou non
- Modèle basé sur la traduction ou non
- Utilisation des mots clés ou d'un d'un dictionnaire
- Datasets utilisés et leurs types
- Les métriques d'évaluation
- Temps d'entrainement
- Représentation des mots

Tableau II-2 Résultats de comparaison des modèles selon différents critères

Modèle / Critère	KCN [49]	Residual LSTM [50]	DGEN [51]	PAMNMT [67]	NSAPAE [69]	VAE-SVG [70]
Langue de paraphrases	Anglais	Anglais	Anglais	Arabe	Arabe	Anglais
Année	2019	2016	2018	2018	2018	2017
Type de réseau de Neurones	RNN bi-directionnel et GRU	LSTM Empilé avec des connexions résiduelles	RNN bi-directionnel	RNN bi-directionnel et RNN	LSTM bi-directionnel	LSTM
Utilisation du mécanisme Attention	Oui	Non	Oui	Oui	Non	Non
Modèle basé sur la traduction	Non	Non	Non	Oui	Non	Non
Utilisation des mots clés	Oui	Non	Non	Non	Non	Non
Utilisation d'un dictionnaire	Non	Non	Oui	Non	Non	Non
Datasets utilizes	MSCOCO PARANMT50M Quora	MSCOCO PPDB WikiAnswers	MSCOCO PPDB Quora	Dataset de métaphores arabes dataset de SIS	EuroParl corpus arabe-arabe	MSCOC Quora
Type de datasets	Monolingue	Monolingue	Monolingue	Bilingue	Bilingue et monolingue	Mono-lingue
Les métriques d'évaluation	BLEU METEOR TER	BLEU METEOR TER Emb Greedy	BLEU METEOR	METEOR cosinus	Cosinus	BLEU METEOR TER
Temps d'en-trainement	N/A	14h MSCOCO 36h PPDB WikiAnswers	N/A	N/A	7 jours	N/A
Représent-ation des mots	Word Embedding	One hot	One hot	N/A	Word Embedding	Word Embedding

Selon les résultats de notre comparaison multicritère, nous pouvons dire que le modèle DGEN [51] utilise un dictionnaire afin d'obtenir des synonymes, toutefois cette solution ne peut pas être appliquée sur la langue arabe faute de ressources (manque de dictionnaires numériques, wordNet, etc.), de même pour le modèle KCN [49] qui, lui, utilise des mots clés à remplacer par leurs synonymes dans la paraphrase.

Concernant les travaux effectués sur la langue arabe, les deux travaux cités dans notre étude se basent sur la traduction comme phase intermédiaire pour la génération de paraphrase.

Al-Raisi et al [69] ont proposé une approche théorique seulement basée sur le Bi-LSTM mais sans résultats concrets. Quant au travail réalisé par Al-Khatib et al [67], il est basé sur un système de traduction, de ce fait l'évaluation a été fait sur la traduction et non pas sur le processus de paraphrase.

Les études ont montré que le Bi- LSTM répond mieux que le Bi-RNN au Traitement Automatique de la Langue, nous allons commencer notre travail par proposer une solution à base de Bi-LSTM au lieu du Bi-RNN contrairement aux travaux [51] et [49], cette solution sera considérée comme solution de référence.

Nous comparerons ces résultats avec deux autres modèles que nous allons développer à savoir l'encodeur/décodeur et l'encodeur/décodeur avec mécanisme d'attention, les plus récentes approches à séquences dans le Deep Learning.

Les résultats obtenus par le meilleur modèle proposé seront comparés avec les travaux étudiés dans ce chapitre.

6 CONCLUSION

Dans ce chapitre nous avons présenté la notion de la paraphrase et l'intérêt d'en générer dans le Traitement Automatique de la Langue, ainsi que toutes les approches existantes pour la génération des paraphrases.

Les plus récentes approches se basent sur les réseaux de neurones approfondis. Cependant les deux travaux arabes cités dans ce chapitre ont utilisé des approches basées sur l'alignement ou la traduction.

Dans le chapitre suivant nous développons notre propre approche basée sur les réseaux de neurones approfondis pour la génération de paraphrase en langue arabe, en essayant d'implémenter plusieurs techniques et d'en choisir la meilleure.

CHAPITRE III : « GENERATEUR DE PARAPHRASES »

1- INTRODUCTION

Notre travail relève de la linguistique informatique, plus précisément de la génération automatique de paraphrases.

Plusieurs travaux connexes existent déjà, les plus prometteurs intègrent le Deep Learning comme outil de génération, mais à notre connaissance aucun n'est dédié à la langue arabe, ce qui fait l'objet de notre travail.

Nous détaillerons dans ce chapitre, l'architecture de notre solution, ses composantes, les différents modèles du Deep Learning proposés, avec les résultats obtenus, pour terminer avec une synthèse d'évaluation.

2- CONTRIBUTION DE NOTRE TRAVAIL

Notre système proposé est un « Générateur de paraphrases ». Son but n'est pas de reproduire de nouvelles phrases équivalentes en soi-même, mais plutôt de l'intégrer dans un système d'évaluation automatique de réponses courtes, lors d'un examen en ligne, ceci permettra d'obtenir une correction automatique plus précise.

Notre contribution est de générer plusieurs réponses modèles automatiquement et décharger l'enseignant de cette tâche. Nous estimons qu'avoir plusieurs réponses modèles permet de prendre en considération toutes les variations possibles dans les réponses des étudiants. Ceci permet entre autres d'améliorer la précision de notation automatique.

3- ARCHITECTURE GLOBALE DE NOTRE SOLUTION PROPOSEE

Notre générateur de réponses modèles est un outil basé sur les réseaux de neurones approfondis. Il permet de générer plusieurs paraphrases à partir d'une seule phrase. Nous avons pour cela

proposé trois modèles Deep Learning.

Dans cette section nous présentons l'architecture de notre système (représentée dans la Figure III.1), adopté pour chacun des modèles proposés, à savoir un modèle Bi-LSTM, codeur-décodeur et un codeur-décodeur avec mécanisme d'attention. Ces derniers ont été entrainés en utilisant les mêmes datasets : le dataset de Al-Raisi [82] pour l'arabe et 'Quora' [83] pour l'anglais. Chaque composante du système est revue plus en profondeur dans les sections qui suivent.

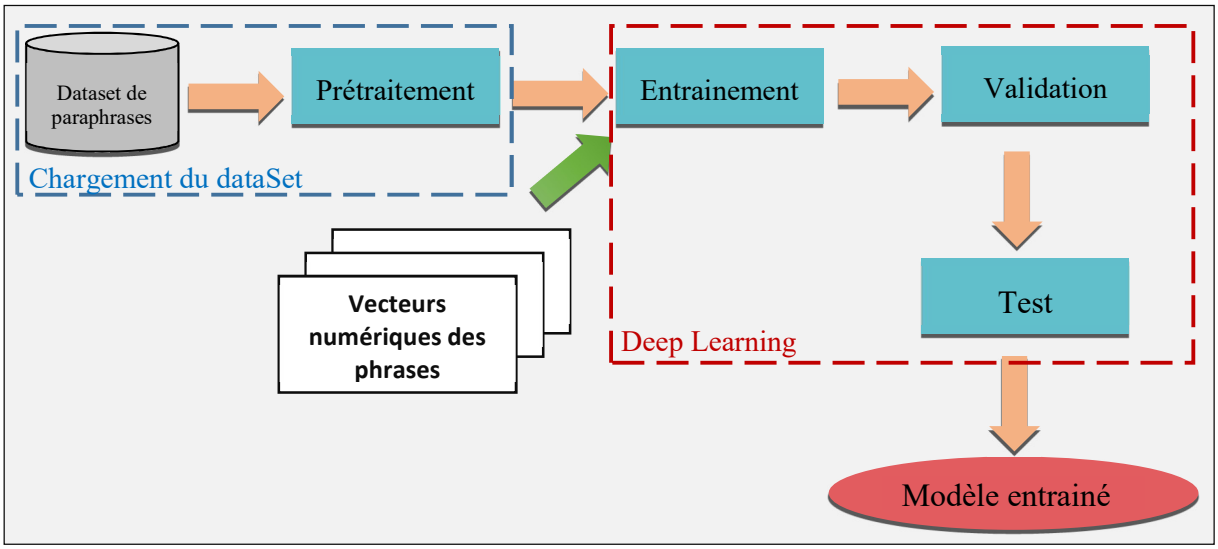


Figure III-1 Architecture globale du Système

Notre système est composé de deux parties principales, la première concerne le chargement et la mise en forme du dataset, et la seconde c'est notre générateur de paraphrases. La sortie de notre système est un ensemble fixe de phrases équivalentes à une phrase courte en entrée.

Puisque tous les modèles proposés reposent sur un entrainement supervisé utilisant un dataset, nous commençons par présenter le prétraitement lié au chargement et à la mise en forme du dataset ; aspect commun à tous les modèles proposés. Nous présenterons ensuite les étapes de notre générateur de paraphrases.

4- CHARGEMENT DU DATASET

Tout système intelligent doit acquérir les connaissances nécessaires pour un meilleur fonctionnement et une meilleure prise de décision. Pour cela, il a besoin de s'alimenter d'un corpus de données (appelé aussi un dataset) ; c'est l'entrée du système.

Dans cette première phase, nous allons lire le datastet utilisé, le nettoyer, le formater, etc., afin

que l'entrée de notre système soit homogène.

La figure III.2 montre en détail le processus du chargement du dataset.

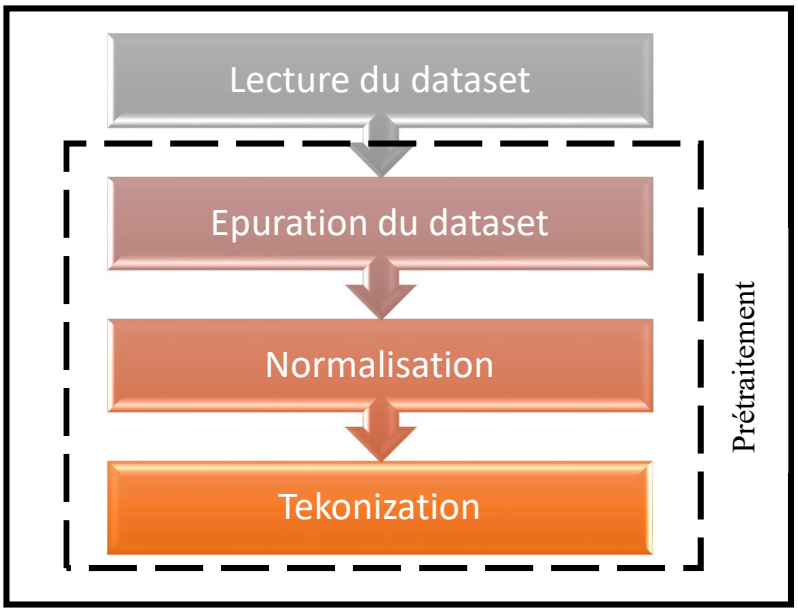


Figure III-2 Processus du chargement du dataset.

4-1 Lecture du dataset

Nous avons conduit nos expérimentations en parallèle pour l'arabe et l'anglais. Nous Avons utilisé donc deux datasets pour l'ensemble de nos modèles.

- Le premier est consacré pour l'arabe, il s'agit du dataset monolingue parallèle réalisé par Fatima Al-raisi. Ce dataset est disponible sur [82], il comporte 200.000 phrases (100.000 phrases et 100.000 phrases équivalentes ou paraphrases) d'une taille d'environ 40,32 Mo. A notre connaissance, ce dataset est le plus volumineux dataset arabe disponible gratuitement, mais nous avons constaté que son contenu comporte plusieurs erreurs et anomalies.
- Le second « Quora », destiné à l'anglais et disponible sur [83]. Il comporte quant à lui 400.000 paires de phrases, il est de taille de 57,93Mo. Ce dataset est parmi ceux qui sont utilisés dans les travaux connexes, ce qui nous permettra de comparer, par la suite, les résultats obtenus.

Les datasets choisis ne sont pas trop volumineux comparant aux exigences de l'apprentissage profond, mais notre choix a été imposé par rapport aux moyens informatiques disponibles.

Par la suite, nous allons consacrer 60% de chaque corpus de données pour l'entrainement, 20% pour la validation, et 20% pour le test comme illustré dans le tableau III.1.

Tableau III-1 Détails d'utilisation des datasets

Dataset	*Entrainement (60%)*	*Validation (20%)*	*Test (20%)*	*Total (100%)*
Quora	210.000	70.000	70.000	350.000
Dataset arabe	46.423	15.474	15.474	77.371

4-2 Le prétraitement

Le prétraitement des données est une étape importante voir cruciale pour le Machine Learning et le Deep Learning. La qualité des données peut directement affecter la capacité d'apprentissage du modèle. Cette étape consiste à transformer des données brutes, en format plus adapté et utilisable par le modèle. Pour cela nous effectuons un certain nombre d'étapes, afin de garantir la qualité des données utilisées par notre modèle :

4-2-1 Epuration des phrases

Cette étape elle-même comporte les opérations suivantes :

a. Nettoyage des datasets

La première concerne le format lui-même du dataset. Nous avons éliminé toutes les données et méta données incorporées dans les datasets non nécessaires pour notre travail. Nous avons gardé seulement les phrases et leurs paraphrases.

b. Elimination des phrases longues

Nous nous intéressons dans notre travail à la correction automatique des réponses courtes, pour cela nous avons éliminé la génération des paraphrases pour des exemples longs (élimination des phrases dépassant n nombre de mots, fixé par l'expert).

Dans notre travail, nous avons fixé la langueur maximale d'une phrase courte à 30 mots que ça soit pour l'arabe ou pour l'anglais.

Cette contrainte a diminué la taille du dataset pour la langue arabe, mais nous a permis quand même d'avoir un dataset plus homogène en nombre de mots, et donc avec des vecteurs d'entiers avec un minimum de 0 insignifiant.

4-2-2 Normalisation

Afin de standardiser le format des phrases, et des mots traités par notre modèle, et afin d'améliorer son efficacité, nous appliquons ce que l'on appelle la normalisation. Cette étape consiste à convertir une liste de mots en une séquence plus uniforme. Lors de cette

normalisation nous avons traité les points suivants :

a- Elimination de la ponctuation

Cette étape consiste à éliminer la ponctuation qu'une phrase peut contenir. Un étudiant ayant saisi dans sa réponse « routeur/commutateur », sera considéré comme étant une réponse incomplète, en la comparant à la réponse modèle « routeur, commutateur ». Il est donc nécessaire de supprimer la ponctuation que ça soit dans les réponses des étudiants que dans la réponse type de l'enseignant.

b- Conversion en minuscule

Appliquée qu'aux langues latines, la mise en minuscule de tous les mots est une étape aussi importante que la précédente, étant donné qu'elle peut affecter la note de l'étudiant vu que l'ordinateur différencie entre la casse.

c- Formater les mots

Une étape qui nécessite certaines transformations permettant à l'ordinateur de reconnaitre les mots similaires, et de ne pas les différencier à cause de leur format. Les changements effectués au niveau des mots, dépendent notamment de la langue. Nous citons ci-dessous quelques exemples de formatage effectués sur les deux langues arabe et anglais.

Exemple pour la langue arabe :

a. Changement de lettre

Avant	Après
إعلام آلي	اعلام الي
سيارة	سياره

b. Elimination de l'accentuation pour la langue arabe « التشكيل »

Avant	Après
الدُّنْيَـا	الدنيا

Exemple pour la langue anglaise :

a. Eclatement des abréviations et sigles

Avant	Après
It's, I'm, we've	it is, I am, we have
US	United States

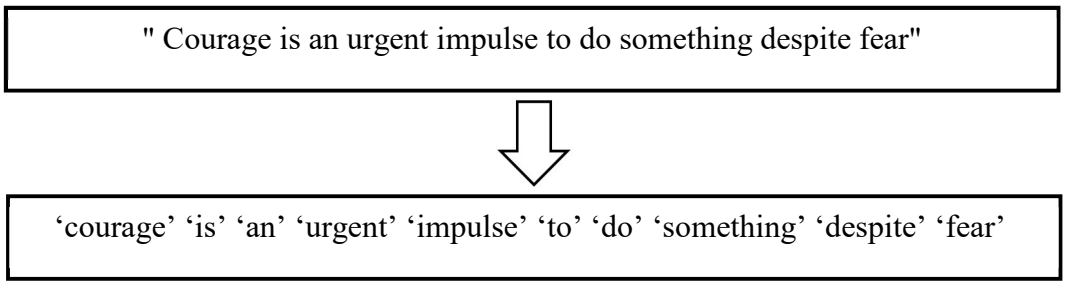
b. Elimination des caractères spéciaux

Avant	Après
e-mail	Email

4-2-3 Tokenisation

Le but principal de la tokenisation est de diviser l'entrée textuelle en un ensemble de fragments, où chaque fragment est appelé jeton. Ces fragments peuvent être des phrases, des mots ou des caractères. Dans notre travail un jeton fait référence à un mot.

Exemple :



Une fois le corpus de données nettoyé et formaté, nous entamerons dans l'étape prochaine les modèles de génération de paraphrases.

5- MODELES « DEEP LEARNING » PROPOSES

Lors de cette deuxième étape, nous entamerons les détails des trois modèles proposés au cours de cette recherche. Il s'agit d'un encodeur-décodeur, un encodeur décodeur intégrant le mécanisme d'attention, et d'un réseau de neurones récurrent Bi-LSTM.

Chacun de ces modèles est exécuté pour les trois phases d'un Deep Learning à savoir : l'entrainement, la validation et le test selon notre architecture proposée.

5-1 Modèle Bi-LSTM

Dans cette section, nous allons expliquer en détails notre premier modèle Bi-LSTM, que nous utilisons comme Baseline. Ce modèle a été proposé vu ses avantages par rapport aux RNN et LSTM, à savoir :

- Il permet la prise en considération de tous les mots de la phrase pour la prédiction d'un mot X. → Bi LSTM, donc ça marche dans les deux sens
- Il permet de traiter des séquences longues.

L'architecture générale de ce modèle est illustrée dans la Figure III.3, où

- ✓ x1, x2, …, x30 représentent la séquence des mots d'une phrase passée en entrée.
- ✓ y1, y2, …, y30 représentent la séquence des mots de la paraphrase générée.

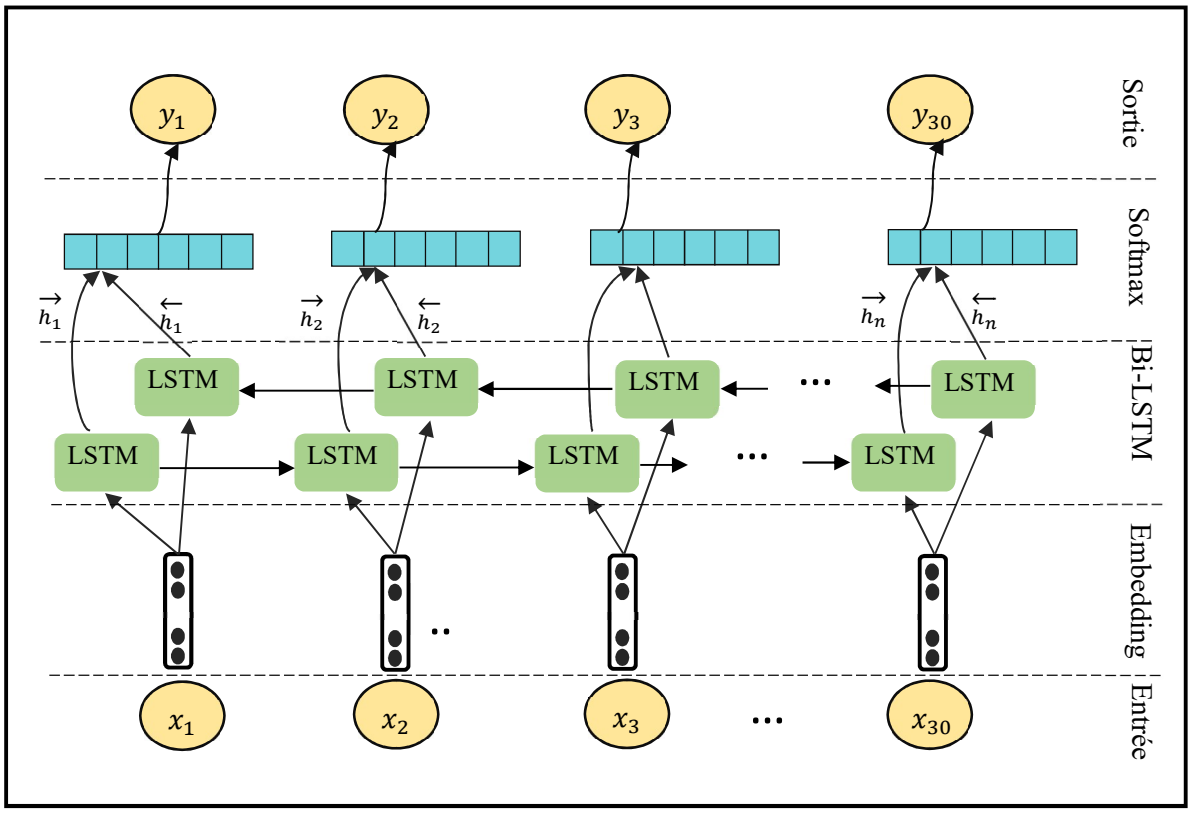


Figure III-3 Architecture globale du modèle proposé Bi-LSTM

Notre architecture est décomposée en trois couches distinctes : la transformation des mots en valeurs numériques (Embedding), la génération d'un ensemble de mots (Bi-LSTM), classification et choix des mots en sortie dont nous détaillons le fonctionnement dans la suite :

5-1-1 Couche « Embedding »

Après avoir effectué tout le prétraitement nécessaire de la première phase, nous obtenons un ensemble de vecteurs de mots, chacun représente une phrase. C'est l'entrée de notre générateur.

Etant donné que les réseaux de neurones approfondis ne peuvent traiter du texte brut, nous devons convertir le texte en vecteurs numériques. Cette conversion appelée « Embedding » dépend du modèle utilisé.

Le Bi-LSTM utilisé dans notre premier modèle nécessite un mapping des données vers des vecteurs en nombres, pour cela plusieurs méthodes existent, nous avons choisi le Word Embedding.

Le Word Embedding est un ensemble de techniques, ayant pour but le mappage de mots en vecteurs de nombres réels, dans un espace dimensionnel réduit [84]. Cette méthode permet de capturer la similarité sémantique, ainsi que le contexte d'un mot dans un corpus donné.

Il existe plusieurs techniques de Word Embedding. Nous nous intéressons au Word2Vec, plus précisément au Skip-gram [84].

Pour l'arabe, nous avons utilisé le Word Embedding de Zahrane [85], qui comporte le mapping d'environ 6,3 million de mots en vecteur d'entier de taille 300.

Concernant l'anglais, nous nous sommes basés sur le Word Embedding faisant parti du référentiel de vecteurs de mots TALL [86]. Il comporte 4027169 mots représenté par des vecteurs de 100.

5-2 Couche « Bi-LSTM »

Les réseaux de neurones Bi-LSTM sont une famille de réseaux de neurones approfondis qui fonctionnent sur des données séquentielles,

Ils se basent sur deux sous couches de LSTM, une couche s'exécutant en avance « Forward » (l'entrée de chaque nœuds LSTM est la sortie nœud traitant le mot précédent) et une autre s'exécutant en arrière « Backward » (l'entrée du chaque nœud LSTM est la sortie du nœud traitant le mot suivant). Les deux états cachés des deux couches combinées sont capables de préserver les informations du passé et du futur.

Chaque modèle Deep Learning y compris le Bi-LSTM, comporte plusieurs paramètres et hyper paramètres considérés comme des éléments caractéristiques à définir, ils peuvent rendre le

modèle beaucoup plus puissant selon les valeurs attribuées.

Les paramètres sont les coefficients que le modèle choisit lui-même tout en étudiant, les optimise au fur et à mesure de l'entrainement, et renvoie un tableau de paramètres minimisant l'erreur.

Par contre, les hyper paramètres, sont à définir par l'administrateur, ils ne seront pas mis à jour par le modèle. Il s'agit de :

- **Fonction d'optimisation** : L'optimisation est la technique qui permet de mettre à jour les paramètres d'un réseau de neurones, elle les converge vers une valeur optimale utilisée pour prédire les labels.

  Nous avons utilisé dans notre modèle l'optimiseur ADAM (Adaptative Moment Estimation) vu ses multiples avantages. Adam permet d'accélérer la convergence par rapport aux méthodes classiques et donc une optimisation dans le temps d'exécution et dans l'utilisation de la mémoire.

- **Fonction de perte (Loss) :** Les réseaux de neurones sont entraînés à l'aide d'un processus d'optimisation, ce dernier nécessite une fonction de perte afin de calculer l'erreur du modèle. Avec les réseaux de neurones, nous cherchons à minimiser l'erreur, plus explicitement, si les prévisions données s'écartent trop des résultats réels, la fonction de perte retourne un très grand nombre. Graduellement, à l'aide d'une fonction d'optimisation, la fonction de perte apprend à réduire l'erreur de prédiction.

  Lors du calcul de l'erreur du modèle pendant le processus d'entrainement, une fonction de perte doit être choisie, dans notre cas, nous avons opté pour la fonction « Categorical CrossEntropy ». Cette dernière est utilisée pour mesurer la distance entre les probabilités des différentes classes et les étiquettes des distributions observées (la probabilité des mots potentiels à générer, par rapport à toutes les classes), la formule mathématique de Categorical Cross Entropy est la suivante :

$$-\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{N} 1_{y_i \in C_c} \log P_{model}[y_i \in C_c]$$

  Tel que :

  - N: est le nombre d'observations.
  - C: nombre de classes.
  - P: la probabilité de l'observation i par rapport à la classe c.

- **Taille des Lots (Batch size) :** Les lots de données sont des petits échantillons de même taille du dataset, sur lesquels l'algorithme s'entraîne au fur et à mesure, cela permettra au modèle de mieux s'adapter aux nouvelles données en estimant le gradient d'erreur avant la mise à jour des poids du modèle. Une époque d'apprentissage signifie que l'algorithme d'apprentissage a effectué un passage sur l'ensemble de données d'apprentissage.
- **Nombre des nœuds LSTM :** Représente le nombre de cellules LSTM utilisées dans une couche cachée, L'idée est d'avoir un réseau de neurone aussi simple que possible mais en même temps qu'il classe bien les données d'entrée.
- **Taux d'apprentissage (Learning rate) :** Le taux d'apprentissage est un hyper-paramètre qui contrôle dans quelle mesure il faut ajuster les poids du réseau par rapport au gradient de perte. En règle générale, le développeur configure le taux d'apprentissage en tirant parti des expériences passées.
- **Taille de l'embedding :** C'est la taille de l'espace vectoriel dans lequel les mots seront incorporés.
- **Taille du buffer :** Le nombre de tuples d'entrainement, (nombre de paires de phrases utilisées).
- **Dropout :** cette technique consiste à supprimer une partie des nœuds qui alimentent une couche, pour que le NN ne soit pas trop lourd. Cela est mis en œuvre pendant la phase d'entrainement. Par exemple, si nous appliquons un dropout égal à 20% sur la couche d'entrées, cela signifie que 1 entrée sur 5 sera exclue de chaque cycle de mise à jour.

Le tableau suivant (III-2) illustre les valeurs attribuées pour chacun des hyper-paramètres.

Tableau III-2 Configuration des hyper-paramètres du modèle Bi-LSTM.

Hyper-paramètres	Fonction d'optimisation	Fonction Loss	Nombre de Nœuds LSTM	Learning rate	Batch Size	Embedding size		Buffer size		DropOut
Définition	Fonction ADAM	Fonction 'Categorical crossEntropy'	256	0.001	16	Ar	300	Ar	46000	0.3
						En	100	En	200000	

5-1-2 Couche « Softmax »

Les résultats de la couche « Bi-LSTM » passent par cette dernière couche, elle agit comme « classifieur », la fonction d'activation Softmax permet d'obtenir une distribution de probabilité sur les mots du vocabulaire. Le mot généré en sortie est celui qui possède la plus grande valeur de probabilité.

5-2 Modèle Encodeur-Décodeur

Les modèles récurrents tel le Bi-LSTM prédisent un mot à partir d'un autre mot (un à un) ou à partir d'une séquence de mots (plusieurs à un), comme ils peuvent prédire un mot à partir d'une séquence de mot (un à plusieurs). Souvent nous sommes confrontés à une situation où l'on doit prédire une séquence de mots directement à partir d'une séquence de mots. Cette forme de prévision est appelée problème de prédiction de séquence de type plusieurs à plusieurs. Pour résoudre ce problème nous proposons comme deuxième modèle l'encodeur/décodeur.

Nous illustrons dans la figure III.4, l'architecture globale de ce modèle. Cette dernière est composée de deux parties distinctes : l'encodeur et le décodeur.

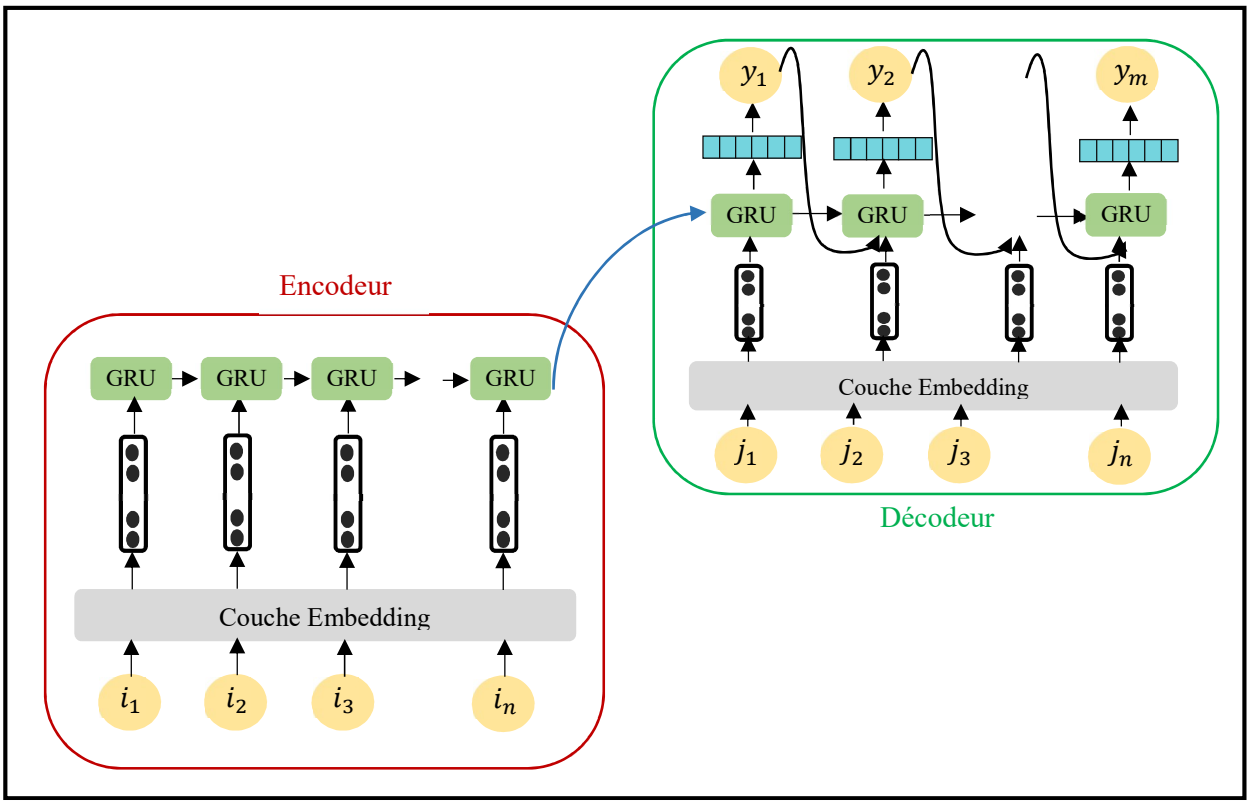


Figure III-4 Architecture globale du modèle proposé Encodeur-Décodeur

Pour ce deuxième modèle, nous optons pour la construction de notre propre Embedding. Ce dernier a comme entrée la représentation numérique (i) de chaque mot (x) de la phrase.

Pour se faire, nous commençons par extraire le vocabulaire des deux datasets arabe et anglais, attribuer une valeur entière séquentielle à chaque mot de notre vocabulaire extrait, le résultat est que chaque phrase est codifiée par un ensemble de numéros. Ce processus est représenté dans la Figure III.5.

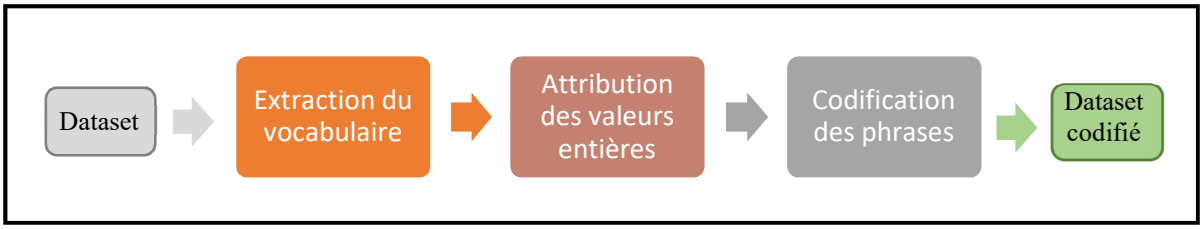


Figure III-5 Processus de représentation numérique du dataset

5-2.1 L'encodeur

Notre encodeur est décomposé en deux modules distincts : l'Embedding, et le générateur du vecteur contenant les informations relatives aux entrées. Ce vecteur est utilisé comme le premier état caché du décodeur, afin de guider le décodeur dans ses prédictions. En ce qui suit, nous détaillerons leurs fonctionnements.

a- Couche Embedding

Dans notre Embedding, nous commençons par représenter chaque mot en vecteur binaire selon l'encodage one hot, puis nous passons par le réseau de neurones qui permettra une représentation réduite tout en gardant les liens sémantiques entre les mots. Un exemple de ce mappage est illustré dans la Figure III.6

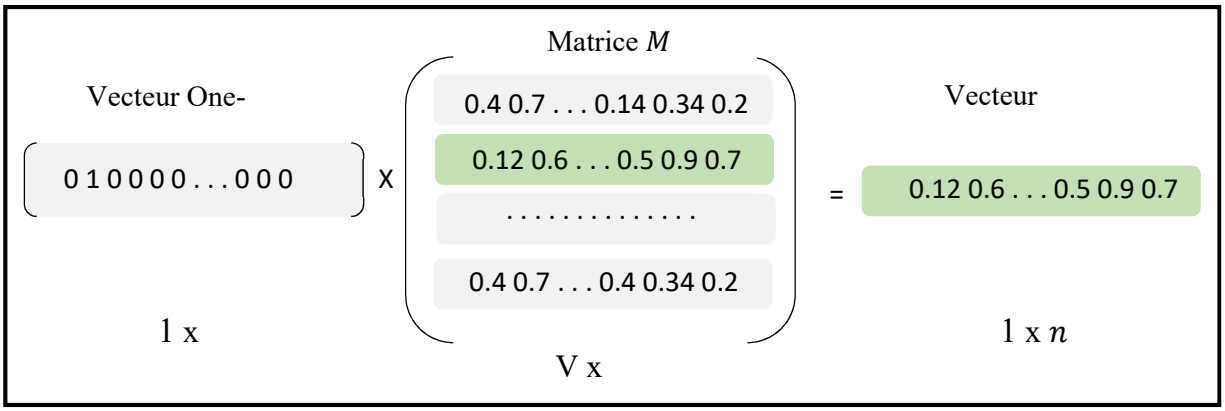


Figure III-6 Exemple d'un mappage de one-hot vers Embedding

Où :

- V : la taille du vocabulaire du dataset
- n: taille du vecteur Embedding
- M: la matrice de pondération (poids), est initialisée avec des valeurs aléatoires et qui sont ajusté lors de l'entrainement.

b- Couche GRU

Apres expérimentation, notre choix est porté sur les cellules GRU vu qu'ils ont montré de meilleurs résultats que ça soit pour le temps de convergence que pour leur efficacité itérative.

Nous illustrons les valeurs des hyper-paramètres utilisés pour ce modèle dans le tableau III.3.

Tableau III-3 Configuration des hyper-paramètres du modèle Encdeur-Décodeur.

Hyper-paramètres	Fonction d'optimisation	Fonction Loss	Nombre de Nœuds GRU	Batch Size	Embedding size	Buffer size	
Définition	Fonction ADAM	Fonction 'Sparse Categorical Cross Entropy'	1024	64	256	Ar	46000
						En	200000

Pour notre Encodeur/décodeur, notre choix pour la fonction Loss est porté sur la fonction 'Sparse Categorical cross entropy'. Cette dernière utilise la même formule que la fonction Categorical Cross Entropy. Le choix entre ces deux fonctions dépend du format dans lequel les sorties du modèle sont représentées.

5-2.2 Le décodeur

Pour notre décodeur, on distingue deux couches, une couche GRU et une couche Softmax.

La couche GRU fonctionne de la même manière que celle de l'encodeur avec une seule exception reposant sur les entrées/sorties :

- Le décodeur prend comme entrée initiale le dernier état caché généré par l'encodeur. Cet état caché contient les informations essentielles contenues dans chaque mot de la phrase d'entrée.
- Tout comme l'encodeur, le décodeur possède une couche Embedding qui, à partir de la représentation numérique de chaque mot de la paraphrase (j_i), génère les vecteurs

Embedding de cette dernière.

- Pour générer un mot y_i au pas de temps t, le décodeur a comme entrée : l'état caché, la sortie générée au pas de temps précédent, ainsi que le vecteur Embedding du i^{éme} mot de la paraphrase.

Pour la couche SoftMax, elle prédit une distribution de probabilité sur les entiers représentants des mots du vocabulaire, et le mot en sortie correspond à l'entier ayant la plus grande probabilité.

5-3 Modèle Encodeur-Décodeur avec mécanisme d'attention

Dans le modèle précédent, le dernier état caché de l'encodeur est un vecteur de longueur fixe contenant les informations nécessaires servant d'entrée au décodeur.

L'inconvénient de cette solution réside dans les séquences d'entrée longues, l'encodeur est incapable de retenir jusqu'à l'état caché final, toutes les informations utiles à la génération de la sortie. Pour résoudre ce problème, nous intégrons au modèle précédent, un mécanisme d'attention tout en gardant la même configuration.

L'intégration du mécanisme d'attention permet de mettre en valeur toutes les informations contenues dans les états cachés aux différents pas de temps au lieu de prendre celles contenues dans le dernier état caché.

La figure III.7 montre l'emplacement du mécanisme d'attention au sein de l'architecture de l'Encodeur Décodeur.

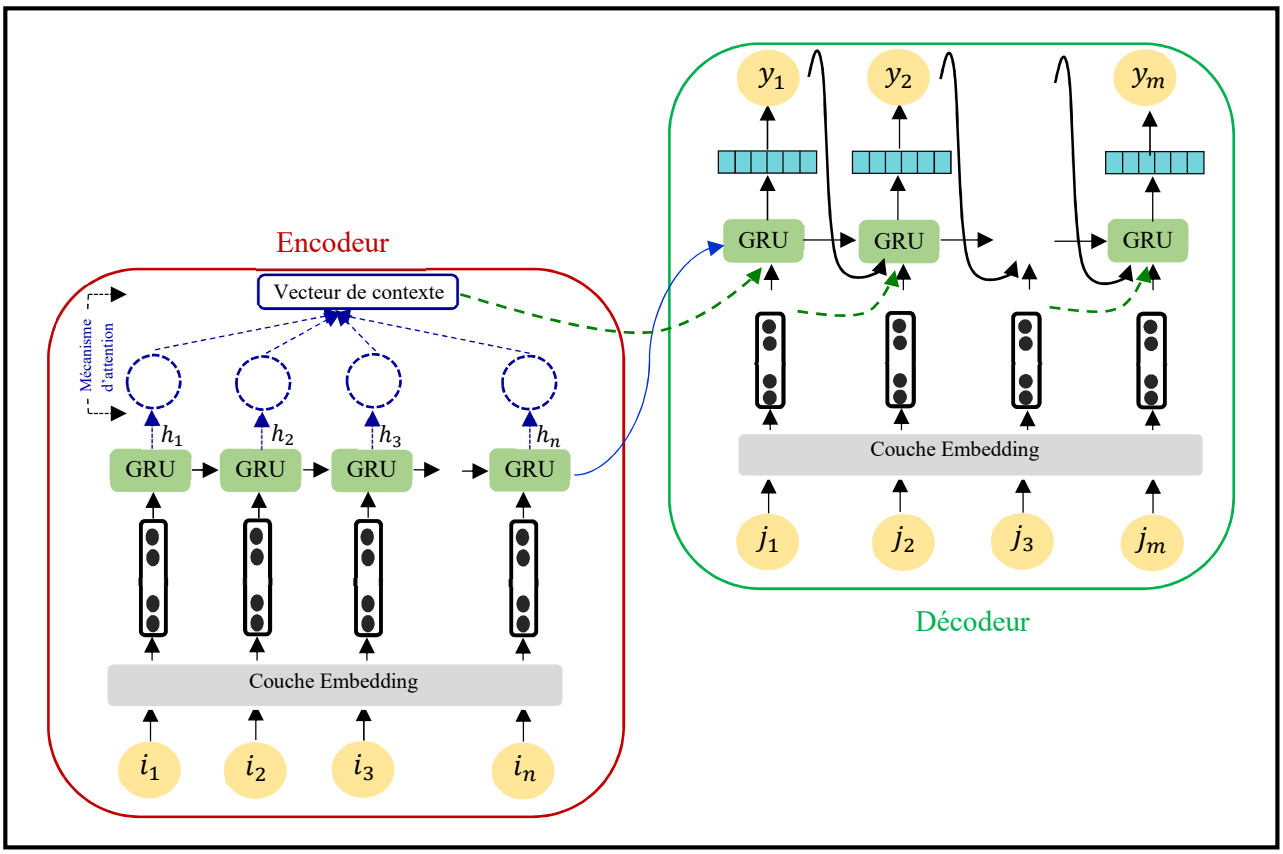


Figure III-7 Architecture globale du modèle proposé Encodeur-Décodeur avec mécanisme d'attention

Pour générer un mot y_i, une attention est accordée à chaque mot de la séquence d'entrée. Elle est exprimée par des poids au niveau de l'encodeur. Ce dernier génère un score pour chaque état caché, de telle sorte que, les états cachés pour lesquels l'attention doit être accordée auront un score élevé.

Les poids d'attention (P_i) sont produits par la fonction softmax, appliquée sur les scores générés par l'encodeur.

Exemple :

Après avoir calculé tous les poids d'attention, un vecteur de contexte est calculé selon la formule suivante :

$$Vecteur\_contexte = \sum_{i=1}^{n} P_i h_i$$

Où :

h_i: Etat caché au pas de temps i.

P_i: Poids accordé à l'état caché h_i

Chaque nœud (GRU_i) du décodeur, hormis le premier, aura comme entrée, la sortie du nœud précédent (y_{i-1}) et le vecteur de contexte généré par le mécanisme d'attention (V_i).

Quant au premier nœud du décodeur (GRU_1), il reçoit en entrée le dernier état caché de l'encodeur avec le premier vecteur de contexte (V_1) généré par le mécanisme d'attention.

6 RESULTATS ET INTERPRETATIONS

Notre travail est initialement dédié à la langue arabe, cependant nous avons jugé opportun d'utiliser l'Anglais comme langue de référence. Ceci nous permettra de comparer nos résultats avec les travaux existants et de valider notre modèle indépendamment de la langue arabe. Cette langue très riche et très vaste n'a pas eu sa part dans les nouvelles technologies de l'information et de communication et dans les nouvelles approches du TAL. Ce qui a induit à un grand manque de ressources numériques pour cette langue.

Chacun de nos trois modèles proposés, est, donc, appliqué pour les deux langues arabe et anglais, nous avons utilisé comme datasets celui de Fatima Al Raisi et Quora, respectivement.

L'entrainement, la validation et les tests de chaque modèle ont été exécutés sur l'environnement Google Colaboratory (proposé par Google Research). Il permet aux développeurs d'implémenter des codes en python. Il est adapté beaucoup plus pour les tâches de Data Science. Cet environnement donne la possibilité d'utiliser un accélérateur graphique GPU afin d'optimiser le temps d'exécution.

Dans cette section nous relevons les résultats obtenus par les métriques d'évaluation automatiques usuelles (BLEU, METEOR, etc.) qui seront vérifiés par une deuxième évaluation qualitative humaine faite par des experts.

Le générateur est évalué de manière intrinsèque par rapport à la notion de génération de paraphrase de manière générale. Une deuxième évaluation extrinsèque du générateur est réalisée d'un point de vue de son intégration à un outil d'évaluation (de notation) automatique des réponses courtes. L'intégration et l'évaluation feront l'objet du prochain chapitre.

6-1 Evaluation intrinsèque du générateur de paraphrases

Une évaluation automatique de nos trois modèles est effectuée en termes de deux métriques BLEU et GLEU pour l'arabe et quatre pour l'anglais à savoir BLEU, GLEU, METEOR et WER

(ces métriques sont définies dans la section II.6 de l'état de l'art).

Tous les résultats obtenus sont résumés dans le tableau III-4. ↑ indique « le plus grand est le meilleur », ↓ indique « Le plus petit est le meilleur ».

Tableau III-4 Résultats obtenus pour les modèles proposés (Bi-LSTM, ED, EDAM) sur l'arabe et l'anglais

Langue	Arabe		Anglais			
Métrique / Modèle	BLEU↑	GLEU↑	BLEU↑	GLEU↑	WER ↓	METEOR ↑
Bi-LSTM	12	04	17	08	9.62	08
Encodeur-décodeur	15	08	19	11	8.89	18
EDAM	**83**	**79**	**54**	**42**	**8.4**	**42**

Le tableau (III-4) regroupe les résultats obtenus pour les langues Arabe et Anglais renvoyés par les métriques d'évaluation automatiques (indiquées en colonne) calculées sur chacun de nos trois modèles proposés présentés en ligne à savoir Bi-LSTM, Encodeur-Décodeur et Encodeur Décodeur avec mécanisme d'Attention (EDAM).

Pour le modèle Bi-LSTM, les résultats obtenus sont très mauvais pour toutes les métriques que ce soir sur l'arabe ou l'anglais. La qualité du word Embedding a influencé négativement sur les résultats obtenus, lors de l'entrainement nous avons constaté que plusieurs mots du dataset arabe et anglais n'avaient pas de représentation en WE. Le modèle les avait remplacés par des zéros non significatifs. D'autant plus, nous estimons que le Bi-LSTM a besoin de plus d'époques lors de la phase d'entrainement pour améliorer ses résultats. Nous le considérons en « baseline » pour pouvoir apprécier les améliorations dans les deux autres modèles.

Une amélioration est constatée pour le modèle Encodeur/Décodeur sur toutes les métriques. Rappelons ici que le modèle prend en compte la construction des Embeddings.

L'ajout du mécanisme d'attention à l'Encodeur Décodeur, a amélioré nettement les résultats obtenus. D'où l'impact important de la considération de l'attention dans le modèle. Ainsi, le troisième modèle proposé (encodeur décodeur avec mécanisme d'attention EDAM), surpasse les deux autres modèles proposés en comparant les scores renvoyés par toutes les métriques utilisées en Arabe et en Anglais.

Le modèle EDAM a donné un bon score BLEU et GLEU pour l'Arabe (83% et 79%

respectivement) comparés aux scores obtenus pour l'Anglais (54%, 42% respectivement). Ceci peut être expliqué par rapport au corpus d'entrainement. Ainsi, la simplicité du dataset facilite au modèle le mappage entre les entrées et les sorties, d'autant plus que le datastet arabe est de petite taille. L'évaluation extrinsèque nous indiquera plus sur la qualité des scores obtenus.

Le décodage de recherche en faisceaux (Beam Search Decoding Algorithm).

L'objectif principal de notre travail est de générer plusieurs paraphrases pour chaque entrée, pour cela nous avons généré avec notre modèle, en premier lieu une seule paraphrase, puis 4, 7 et finalement 10 en utilisant l'algorithme de décodage (pour le décodeur). Cet algorithme utilise une recherche en faisceaux (Beam Search Decoding Algorithm) [87], où le décodage de la séquence de sortie la plus probable implique une recherche dans toutes les séquences de sortie possibles en fonction de leur probabilité. Ainsi plusieurs paraphrases sont générées en fonction de la taille du faisceau (que nous appelons dans la suite le Beam).

Dans le tableau III-5, nous exposons les résultats obtenus par l'évaluation automatique de la métrique BLEU sur le dataset de la langue arabe, de notre modèle EDAM-Ar avec une génération d'une seule paraphrase en sortie, de 4, de 7 et de 10 paraphrases. La génération de plusieurs paraphrases nous permet d'introduire les nouvelles métriques (Avg-Bleu & Best-Bleu) qui introduisent respectivement la moyenne et le meilleur Bleu.

Tableau III-5 Résultats obtenus pour la génération de multiples paraphrases (1, 4, 7 et 10) par notre modèle EDAM-Ar.

Modèle	Beam	Avg-Bleu	Best-Bleu
EDAM-Ar	Beam 1	83	N/A
	Beam 4	59	N/A
	Beam 7	55	N/A
	Beam10	53	54

Chaque ligne du tableau ci-dessus représente le résultat obtenu par rapport au nombre de paraphrases générées (les Beams) par notre modèle EDAM-Ar, selon la métrique BLEU. Nous avons calculé pour chaque ligne la moyenne des résultats (AVG- BLEU) et le meilleur résultat (BEST-BLEU) pour le beam = 10.

AVG-BLEU. Pour calculer le score AVG-BLEU, nous calculons le score BLEU entre chaque couple. Ensuite la moyenne des moyennes est calculée. (phrase générée, paraphrase (référence)), puis nous appliquons la moyenne.

Rappelons ici qu'en entrée du modèle nous avons des couples d'entrainement du dataset (Phrase source, Phrase de référence). La phrase générée est la sortie produite par le modèle entrainé.

Nous constatons, par rapport à la moyenne BLEU, que le score calculé par cette métrique diminue à chaque fois que le nombre de paraphrases générées augmente (83% pour Beam=1 jusqu'à 53% pour Beam=10). Ceci est prévisible dans le sens où le calcul de la moyenne inclut le cas meilleur et le cas pire. Moyennant aussi avec Beam = 1, la fonction softmax prévoit la plus grande probabilité directement pour l'unique paraphrase générée.

Selon le tableau III-6 comportant l'interprétation des scores BLEU repris de l'évaluation des modèles proposés par Google [88], nous pouvons dire que les toutes les paraphrases générées de haute qualité (Pour Beam= 10 nous avons eu un score de 53%).

Tableau III-6 Interprétation des scores BLEU [88]

Score Bleu (%)	Interprétation
<10	Résultat presque inutile
10 à 19	L'idée générale est difficilement compréhensible
20 à 29	L'idée générale apparait clairement, mais le texte comporte de nombreuses erreurs grammaticales
30 à 40	Résultats compréhensibles et correctes.
40 à 50	Résultat de haute qualité
50 à 60	Résultat de très haute qualité, adéquat et fluide
> 60	Qualité souvent meilleure que celle donnée par l'humain.

Best-Bleu. Est une métrique calculée lors d'une génération de plusieurs paraphrases. Nous l'avons utilisée pour Beam=10 à des fins comparatives avec d'autres modèles de la littérature. Nous constatons que le score renvoyé est très bon selon l'interprétation de BLEU.

Pour obtenir le score Best-BLEU, nous calculons la métrique BLEU entre chaque phrase d'entrée (source) d'une part, et chacune des phrases générées (les 4, 7 ou les 10) associées à cette dernière d'une autre part, et ce, pour obtenir la meilleure variante, puis rapportons les scores BLEU à partir du corpus contenant les paraphrases de références (les phrases en sortie dans le dataset) et les meilleures variantes obtenues lors de la dernière étape.

Les résultats obtenus par cette évaluation automatique seront complétés avec une deuxième évaluation qualitative. Celle-ci sera effectuée par des experts de la langue, cela nous permettra de vérifier les scores obtenus.

Comparaison avec les résultats de la littérature. Nous avons par la suite, entrainé notre modèle EDAM sur le dataset Quora pour l'anglais, et nous l'avons évalué avec les métriques les plus usuelles de la littérature à savoir BLEU et METEOR, les résultats obtenus sont illustrés dans le tableau III-7 comparés avec deux travaux connexes récents ayant généré dix paraphrases pour le même corpus de données (i.e Quora).

Tableau III-7 Résultats EDAM Vs Littérature sur le dataset Quora .

	Beam =1		***Beam=10***			
	Moyenne (AVG)		*Moyenne (AVG)*		*Best BLEU*	
Métrique / **Modèle**	Bleu	Meteor	Bleu	Meteor	BLEU	METEOR
VAE-SVG-eq [70] **(2017)**	26.2	25.7	37.1	**32**	38	**32.9**
Ours (EDAM-En)	54.0	42	43.0	24	**49**	28
GAP [89]**(2019)**					47.6	30.94

Le tableau est constitué de deux grandes parties, la première englobe les résultats de l'évaluation pour la génération d'une seule paraphrase (Beam=1) par modèle. Nous avons utilisé pour cela la Moyenne de BLEU et la Moyenne de METEOR. La deuxième partie du tableau comporte l'évaluation pour un nombre de paraphrases générées égal à dix (Beam =10) par modèle.

Les résultats obtenus pour chacun des trois modèles représentés à savoir VAE-SVG-eq, EDAM-En et GAP sont représentée en ligne.

Pour la génération d'une seule paraphrase à la fois, le modèle GAP ne rapporte aucune évaluation, il a été conçu pour générer plusieurs paraphrases à la fois seulement.

Comparant notre travail avec VAE-SVG-eq, nous constatons que le nôtre le surpasse largement, selon les résultats obtenus par BLEU et METEOR (54.0 contre 26.2 selon Bleu et 42.0 contre 25.7 selon METEOR).

En se référant aux scores obtenus et au tableau d'interprétation (III-6), nous pouvons dire que les paraphrases générées par EDAM-En sont de haute qualité tandis que VAE-SVG-eq génère

des paraphrases comportent de nombreuses erreurs grammaticales.

Lors de la génération de plusieurs paraphrases à la fois, le tableau III-7 montre que le résultat obtenu reste toujours de bonne qualité (AVG BLEU=43 et BEST-BLEU=49) pour notre Modèle EDAM-En, d'autant plus, selon ces deux métriques de BLEU, notre modèle donne de meilleures paraphrases par rapport à GAP (BEST-BLEU=47,6) et VAE-SVG-eq (AVG BLEU=37.1 et BEST-BLEU=38).

6-2 Evaluation humaine qualitative.

Les évaluations automatiques fournissent des mesures utiles sur la qualité du langage (grammaire et forme). Cependant, elles ne fournissent aucune mesure sur la pertinence du contenu obtenu (la sémantique) [90]. C'est pourquoi des évaluations basées sur des jugements humains sont complémentaires aux évaluations des métriques automatiques.

Dans cette optique, une deuxième évaluation manuelle a été demandée à des experts en arabe et en anglais sur un échantillon de 100 paraphrases générées du jeu de test.

Cette évaluation, effectuée dans une perspective fine de vérifier deux aspects fondamentaux difficiles à évaluer avec Bleu, Meteor, … C'est deux aspects sont liés à :

- ✓ **La « Relevance »** (Pertinence en sens) : exprime la pertinence de la paraphrase générée avec la phrase d'entrée. Ici il est question de noter à quel point la phrase générée préserve le même sens que la phrase originale.
- ✓ **La « Readability »** (lisibilité en forme) : la lisibilité de la paraphrase générée en termes de forme, de grammaire sans considérer le sens de la phrase générée.

Nous avons retenu ces deux aspects parce que nous disposons de ces valeurs pour le dataset Quora lors de son élaboration et des valeurs pour le travail de VAE-SVG-eq.

Pour le dataset arabe, aucune information sur une évaluation qualitative n'est disponible.

Pour chacun des deux aspects, un annotateur attribue un score compris entre 1 et 5, où 1 est le pire et 5 est le meilleur.

Les annotations sont données par trois experts pour chacune des deux langues sur l'ensemble de l'échantillon arabe et anglais.

Nous exposons dans le tableau III-8 la moyenne des annotations obtenues pour nos modèles retenus EDAM-Ar et EDAM-En, les annotations du dataset Quora, ainsi que le modèle VAE-

SVG-eq.

Tableau III-8 Annotations humaines qualitatives

Modèle	Relevance	Readability
Quora Dataset	4.82	4.94
Quora Dataset (VAE-SVG-eq)	3.57	4.08
Quora Dataset (EDAM-En)	**3.58**	**4.51**
Al Raisi Dataset (EDAM-Ar)	**3.52**	**3.88**
Al Raisi Dataset	N/A	N/A

D'après l'évaluation humaine rapportée dans l'article de Quora [70], les experts ont estimé que les paraphrases que contient ce dataset ne sont pas exactes à 100%, ils estiment que la pertinence du sens est à 4.82/5 ce qui signifie environ 96.4% du sens est préservé en moyenne dans tout le corpus de génération des paraphrases, avec une lisibilité textuelle (grammaire et forme) dépassant 98% (4.94/5) ce qui nous permet de dire que le corpus est de très bonne qualité que ce soit forme ou contenu.

Quant aux modèles nous pouvons déduire, selon le tableau III-8, pour l'anglais, que la pertinence des phrases générées automatiquement par notre modèle EDAM-En et VAE-SVG-eq est estimée en moyenne 71.6% et 71,4% respectivement ce qui nous permet de dire que la sémantique des paraphrases est assez préservée globalement. Cependant, les phrases générées par notre modèle sont plus correctes syntaxiquement que le modèle de référence avec une lisibilité grammaticale moyenne de 90.2% contre 81.6% pour le modèle VAE-SVG-eq.

Nous constatons que les annotations des experts arabes sont assez bonnes (70.4% du sens est préservé et la lisibilité des paraphrases généré est à 77,6%).

Nous présentons ci-après deux exemples de paraphrases générées par notre modèle EDAM, l'un en arabe et l'autre en anglais.

Phrase origine	السيد الرئيس، اود ان ابدا بتوجيه الشكر إلى جميع زملائي الاعضاء في البرلمان وفي اللجنة على الدعم الذي قدموه لي في عملي بشان هذا التقرير
Phrases référence	السيد الرئيس اود ان ابدا بشكر جميع زملائي في البرلمان واللجنه على دعمهم في هذا التقرير
Phrase générée par EDAM-Ar	السيد الرئيس اود ان ابدا بتوجيه الشكر الى جميع زملائي في البرلمان واللجنه على دعمهم في هذا التقرير

Phrase origine	why do some people ask questions on quora that could be asked directly to a search engine
Phrases référence	why don't many people posting questions on quora check Google first
Phrase générée par EDAM-En	why do people ask questions on quora that could simply be googled

En conclusion, ces annotations manuelles des experts confirment les bons résultats obtenus par l'évaluation automatique, d'autant plus il y a globalement une bonne corrélation entre les annotateurs d'EDAM selon le tableau suivant (III-9).

Tableau III-9 Corrélation entre les annotations des experts en Anglais et en Arabe selon le coefficient de Pearson

Métriques d'évaluation	**Anglais**			**Arabe**
	Expert 1 – Expert 2	**Expert 1 – Expert 3**	**Expert 2 – Expert 3**	**Expert 1- Expert 2**
Corrélation Relevance	0,68089859	0,57168245	0,46173883	0,64152363
Corrélation Readability	0,6779913	0,6591057	0,6277413	0,511621531

Si nous devons considérer uniquement les deux experts qui ont le plus corrélé ensembles dans les deux aspects (à savoir expert 1 et expert 2), alors la Readability = 4,48 et la relevance = 3,58. Ce sont encore des valeurs comparables avec la moyenne entre les 3 experts et confirment bien le résultat obtenu.

7 CONCLUSION

Dans ce chapitre, nous avons proposé trois modèles de génération de paraphrases basés sur les approches du Deep Learning, à savoir un modèle LSTM Bidirectionnel (Bi-LSTM), un modèle Encodeur-Décodeur (ED) et un troisième modèle Encodeur Décodeur avec Mécanisme d'Attention (EDAM) dans le but de combiner les avantages des ED avec le MA.

Nous avons entrainé les trois modèles pour deux datastets distincts, l'un en arabe et l'autre en anglais.

Les résultats expérimentaux évalués sur des métriques automatiques démontrent que le dernier modèle proposé à savoir EDAM apporte les meilleurs scores par rapport aux deux modèles proposés mais aussi aux travaux de la littérature. Des évaluations humaines ont vérifié également l'efficacité de notre modèle.

Notons qu'il nous a été impossible de comparer nos résultats pour la langue arabe vis-à-vis de la littérature, puisque, aucun travail traitant de l'arabe ne présente une évaluation concrète aussi bien quantitative (Bleau, Meteor, ...) que qualitative (Relevance, Readability). Nous nous sommes fiés aux indicateurs des métriques elles-mêmes pour évaluer nos modèles.

Dans le chapitre suivant, nous allons intégrer notre modèle EDAM-Ar à une système d'évaluation automatique des réponses courtes (notation d'examen) pour voir l'impact impliqué par l'utilisation de plusieurs réponses modèles.

CHAPITRE IV : INTEGRATION DE EDAM AU SYSTEME D'EVALUATION AUTOMATIQUE

1- INTRODUCTION

Dans l'éducation, l'évaluation est un processus visant à mesurer le niveau d'apprentissage, qui est capital dans tout processus d'enseignement.

Elle permet de positionner le sujet évalué sur une échelle de progression, et de savoir ainsi ce qu'il reste à faire pour arriver au but, conduisant à reformuler les connaissances, pour les assimiler et les appliquer d'une manière plus satisfaisante.

Cependant, l'évaluation est un processus couteux en temps et très difficile à élaborer par l'enseignant. Afin de soutenir les tâches pédagogiques de l'enseignant et minimiser les erreurs humaines dues à plusieurs paramètres telle la subjectivité, la fatigue, la surcharge de correction, etc., des solutions de correction automatique ont vu le jour, mais majoritairement dédiées aux réponses de QCM (questions à choix multiples), quant aux questions ouvertes, les travaux existants n'ont pas prouvé leurs efficacités malgré les progrès technologiques.

L'objectif principal de notre travail, est de proposer un générateur automatique de paraphrases à partir de réponses types (corrigé type donné par l'enseignant), d'une épreuve (un examen) et de l'intégrer à un système d'évaluation automatique de réponses, afin d'améliorer la qualité de la correction automatique des examens à questions ouvertes.

Pour cela, nous proposons d'intégrer notre générateur au sein d'un système d'évaluation automatique de réponses courtes à des questions ouvertes. Le système utilisera l'ensemble des paraphrases générées comme un ensemble de réponses modèles qui servira dans son processus de correction automatique.

Notre proposition est basée sur le principe que l'étudiant pourra exprimer sa réponse de

plusieurs manières, tout en gardant la même idée que la réponse type de l'enseignant, afin de ne pas pénaliser l'étudiant et d'avoir plus de chance à faire une correction automatique performante. Nous estimons que comparer la réponse de l'étudiant avec plusieurs réponses types à la même question permettra au système automatique une meilleure précision.

Dans ce chapitre nous allons illustrer l'architecture du système d'évaluation automatique que nous cherchons à améliorer, et montrer l'emplacement de notre générateur dans cette architecture globale.

Par la suite, nous passons à la partie expérimentale où, nous illustrons nos jeux d'essai utilisés dans nos tests, et les métriques d'évaluation quantitative et qualitative. Nous terminons par exposer et discuter les résultats obtenus et examiner l'impact de notre générateur sur la précision de la notation

2- ARCHITECTURE GLOBALE DU SYSTEME D'EVALUATION AUTOMATIQUE

Le Système d'évaluation automatique global dont nous cherchons à améliorer les performances, est un plugin proposé par A.Madani et E.Snoussi [91] dédié à la langue Arabe. Il a été intégré au sein de la plateforme de e-learning Moodle [92].

Le système est composé initialement de deux modules principaux :

- Un gestionnaire de test : est un composant propose de Moodle, permettant à l'enseignant d'introduire un examen, le corrigé type et le système de notation.
- Un évaluateur automatique : est un système qui permet d'évaluer la réponse d'un étudiant à la réponse modèle donnée par l'enseignant, et de renvoyer un score (ce score reflète la note que l'étudiant a eu pour sa réponse).

Nous avons rajouté un troisième module à l'architecture, il s'agit de notre générateur automatique de paraphrases EDAM, qui sera intégré au système, il interagira avec l'évaluateur automatique en lui fournissant plusieurs paraphrases équivalentes à la réponse modèle de l'enseignant. L'évaluateur corrigera, donc, la réponse de l'étudiant en la comparant à toutes les réponses modèles similaires et gardera le meilleur score (la meilleure note pour la réponse donnée par l'étudiant).

Nous illustrons dans la figure IV-1 une architecture simplifiée du nouveau système d'évaluation automatique, où nous montrons l'interaction entre les différents modules du système.

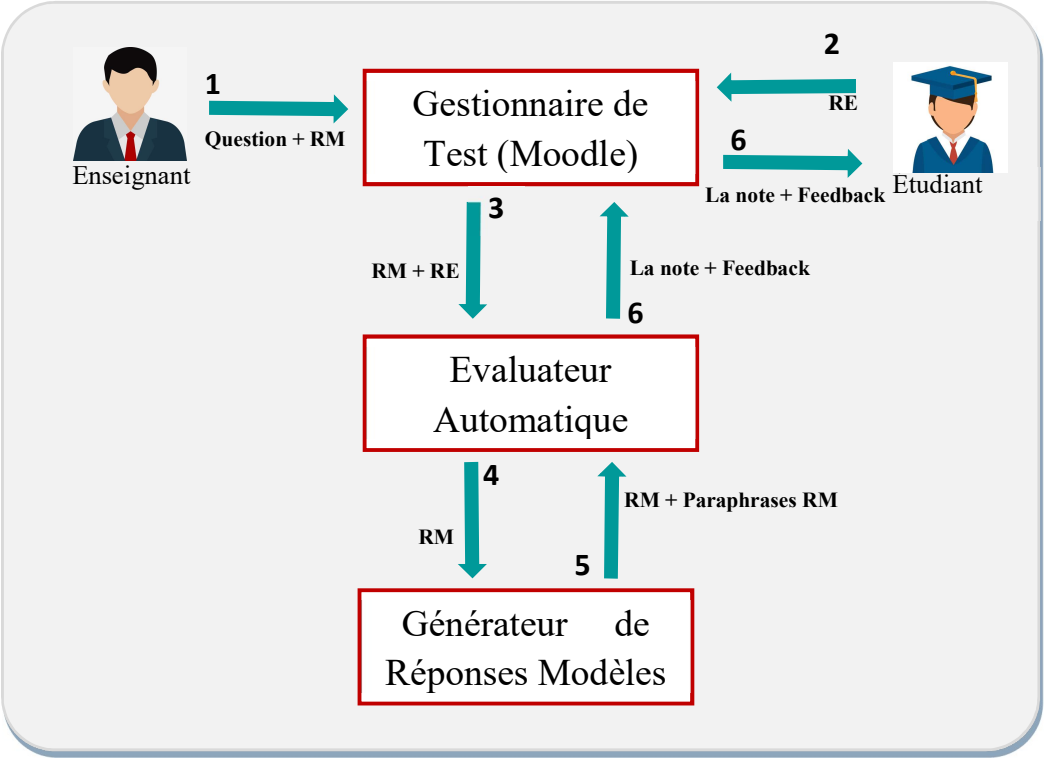


Figure IV-1 Architecture globale du système d'évaluation global.

RM : Réponse Modèle et RE : Réponse Étudiant). Comme ça le schéma parle de lui-même. Dans ce système d'évaluation automatique des réponses(ASAG), deux acteurs principaux interviennent à savoir l'enseignant et l'étudiant.

Le processus de l'évaluation est lancé une fois l'enseignant programme une épreuve sur la plateforme Moodle. Avant d'expliquer en détail les différents modules de ce processus, nous expliqueront d'abord dans le scénario qui suit le fonctionnement du processus :

1. **Déposer l'épreuve, le corrigé type et le barème :** L'enseignant saisira sur la plateforme de Moodle l'ensemble des questions ouvertes de l'épreuve, la réponse modèle (RM) de chaque question et son barème.
2. **Répondre aux questions :** L'étudiant répondra aux différentes questions de l'épreuve (RE) dans un espace de temps limité.
3. **Lancer la correction automatique :** Le gestionnaire de test invoquera, une fois les réponses de l'étudiant (RE) et la réponse modèle (RM) enregistrées, le module de l'évaluation automatique afin de lancer la correction automatique des réponses et calculer

la note.

4. **Invoquer le générateur de paraphrases :** Afin de pouvoir évaluer la réponse de l'étudiant, l'évaluateur automatique fait appel au générateur de paraphrases, en lui transmettant la réponse modèle (RM). En effet, la correction automatique nécessite la réponse modèle de l'enseignant (RM), la réponse de l'étudiant, le barème ainsi que la liste des paraphrases récupérée.
5. **Générer les réponses modèles :** Le générateur automatique de paraphrases EDAM renvoie à l'évaluateur l'ensemble de paraphrases similaires aux réponses modèles.
6. **Afficher la note de l'étudiant :** Une fois la note globale calculée, elle sera affichée, via le gestionnaire de test.

En ce qui suit, nous détaillerons le fonctionnement de chaque module du système automatique à savoir le gestionnaire de test, l'évaluateur automatique et EDAM.

2-1 Module 1 : (Gestionnaire de Test 'Moodle')

Moodle est une plateforme d'E-learning qui consiste à utiliser les ressources de l'informatique et de l'Internet, afin d'acquérir des connaissances en accédant à distance à des cours.

Nous avons choisi Moodle afin d'intégrer notre système d'évaluation automatique, sachant que c'est une plateforme internationale, open source et gratuite permettant d'intégrer de nouveaux modules.

Moodle comporte plusieurs fonctionnalités dont : La diffusion des cours accessibles à tout moment aux étudiants, proposition des devoirs en ligne, effectuation des tests d'auto-évaluation…

Nous nous intéressons au gestionnaire de test de Moodle. Ce module permet à l'enseignant de concevoir et de créer des questionnaires constitués d'une grande variété de Types de questions, y compris les types de questions à choix multiple, les vrai-faux, les réponses courtes et le glisser-déposer d'images et de texte. Ces questions sont conservées dans la Banque de questions et peuvent être réutilisées dans différents questionnaires.

Toutefois il ne comporte pas d'évaluateur automatique des réponses pour tous les types de questions. Aucune solution n'est proposée pour les réponses courtes formulées en langage naturel. Cette limite fait l'objet du projet de recherche dont s'intègre notre travail.

2-2 Module 2: (ASAG)

Le module d'évaluation automatique représente la composante principale du système, il permet d'attribuer un score à la réponse de l'étudiant en la comparant avec la réponse modèle. Le principe général est de calculer le degré de similarité entre les deux réponses.

Nous avons combiné notre générateur à deux modèles du système d'évaluation ASAG :

- ASAG V1.(2018/2019) [91] développé en se basant sur des approches statistiques basées corpus combinant des similarités syntaxiques et sémantiques utilisant des espaces sémantiques.
- ASAG V2.(2019/2020) [93] Développé en utilisant un modèle de Machine Learning. Cette nouvelle version améliore la précision de la V1.

Dans le cadre de notre travail, nous omettons de donner des détails sur les outils ASAG déjà développés (ASAG V1. & ASAG V2), car nous avons utilisé directement leurs codes déjà développés et pour ne pas sortir de notre objectif qui est l'intégration du générateur pour considérer plusieurs réponses modèles.

L'aspect modulaire de l'ASAG nous a permis de comprendre et de délimiter facilement notre intervention.

En introduisant le générateur de paraphrase, chaque réponse modèle permet de générer plusieurs réponses modèles différentes.

L'évaluateur calcule la similarité entre la RE et chacune des paraphrases et considère le score max qui est renvoyé à l'étudiant avec le feedback correspondant.

Dans l'Annexe B, le détail des algorithmes de l'évaluateur automatique ASAG est donné.

2-2 Générateur de réponses modèles

Comme expliqué dans le chapitre précédent, le générateur des réponses modèle EDAM, que nous avons développé, est un outil basé sur le Deep Learning. Il permet de générer plusieurs paraphrases à partir d'une réponse modèle.

3- EXPERIMENTATION

Pour la correction automatique, afin de tester l'utilité de comparer plusieurs réponses modèles aux réponses des étudiants, et voir l'impact de notre proposition sur la précision des résultats obtenus, nous examinerons les évaluateurs automatiques (cités dans la section 3-2), avec les paraphrases générées par notre solution EDAM sur deux jeux d'essaies, l'un est en anglais et l'autre est en arabe.

Lors des tests, nous évaluerons les résultats obtenus en les comparants avec les notes des enseignants. Nous utiliserons comme métriques le coefficient de corrélation de Pearson et l'erreur quadratique moyenne (RMSE) que nous présenterons par la suite.

En ce qui suit, nous définissons brièvement les deux jeux d'essais arabe et anglais pour les outils ASAG :

3-1 Jeux d'essais pour les outils ASAG

Nous utilisons le dataset « Mohler DS » [94] comme jeu d'essai pour l'anglais, ce dernier est un dataset largement cité dans l'évaluation automatique qui est disponible.

Nous optons pour le jeu d'essai arabe à celui réalisé par [95] portant sur un examen de la Cybercriminalité.

3-1-1 Dataset de Mohler « Mohler DS »

Le jeu d'essai de Mohler comporte trois sujets d'examens portant sur un cours d'initiation à l'informatique de l'Université de North Texas [94], [96].

Chaque sujet comprend l'ensemble des questions, des réponses types de l'enseignant, des réponses des étudiants avec les notes moyennes de deux annotateurs.

Les notes sont comprises entre 0 et 5. La corrélation entre les annotateurs sur l'ensemble des données était de 0,6443 selon le coefficient de Pearson.

Le jeu d'essai de Mohler est composé de 630 réponses pour 21 questions au total, la répartition des réponses est illustrée dans le tableau IV-1.

Tableau IV-1 Contenu du jeu d'essai Mohler

Numéro de sujet	Nombre de questions avec réponse modèle	Nombre de réponses par question	Total de réponses
1	7	29	203
2	7	30	210
3	7	31	217

Chacun des trois sujets comporte 7 questions avec leurs réponses modèles, pour le 1er sujets, la réponse de 29 étudiants a été enregistrée pour chaque question, pour le 2ème sujet, il y a eu 30 réponses par question. Quant au dernier sujet 31 réponses par question ont été enregistrées, ce qui fait un total de 630 réponses.

Le tableau IV-2 est un échantillon du datastet comportant une question de chaque sujet, sa réponse modèle, la réponse de deux étudiants et la note moyenne attribuée par les annotateurs.

Tableau IV-2 Un échantillon du jeu d'essai Mohler

ID Question	Question	Réponse Modèle	Réponse Étudiant	Note Manuelle
Question 1 Assign 1	What is the role of a prototype program in problem solving?	To simulate the behaviour of portions of the desired software product	It tests the main function of the program while leaving out the finer details. 	2
			it simulates the behavior of portions of the desired software product	5
Question 1 Assign 2	What is typically included in a class definition?	Data members (attributes) and member functions.	An object and data.	2
			Data and functions	4.5
Question 1 Assign 3	What does a function signature include?	The name of the function and the types of the parameters.	input parameters and return type	3
			The portion of the function prototyp tha has the function name and the arguments but NOT the return type.	5

Nous avons généré dix paraphrases pour chaque réponse modèle du datastet « Mohler DS », ce qui nous a donné un ensemble de 210 paraphrases générées au total.

3-1-2 Dataset arabe « Cyber criminality Arabic Short Answer Grading Data set (CCASAG) »

Le jeu d'essai arabe CCASAG est un dataset comportant 3 sujets, chacun contenant 18 questions portant sur la cybercriminalité, il a été effectué à l'université Blida 1, il contient 48 questions non répétées, avec un nombre total de 2133 réponses, la note attribuée à chaque réponse reflète la moyenne de deux notes calculées par deux enseignants.

Tout comme le jeu d'essai Mohler, les notes sont comprises entre 0 et 5. La corrélation entre les annotateurs sur l'ensemble des données était de 0,83 selon le coefficient de Pearson[97],[98].

Nous illustrons dans le tableau IV-3 un petit échantillon du dataset arabe « CCASAG »

Tableau IV-3 Un échantillon du jeu d'essai (CCASAG)

ID Question	Question	Réponse Modèle	Réponse Étudiant	Note Manuelle
Question 1	عرف مصطلح الجريمة الإلكترونية	سلوك قانوني يتم باستخدام الاجهزه الالكترونيه الهاتف الكمبيوتر الانترنت ينتج حصول المجرم فوائد ماديه معنويه تحميل الضحيه خساره وغالبا هدف الجرائم القرصنه سرقه اتلاف المعلومات وتكون عاده الانترنت اداه مسرحا	سلوك اخلاقي يتمطريق وسائل الكترونيه يهدف عائدات ماديه يسبب اضرارا للضحيه	3
			سلوك قانوني تستخدم الوسائل الالكترونيه ينتج حصول المجرم اهداف ماديه معنويه هدف الجرائم القرصنه سرقه اتلاف المعلومات	4
Question 9	ما نوع الجريمة الإلكترونية التي قد تكون أنت ضحيتها عبر بريد إلكتروني يخبرك بأن شخصا يريد أن يتقاسم معك جوائز أو أموالا؟	جرائم النصب والاحتيال تصنف المجموعه الاولي تقع الانترنت الشبكه محلا	الجريمه تصنف الهندسه الاجتماعيه النفسيه البريد الالكتروني	0,87
			النصب والاحتيال والابتزاز جرائم تقع الانترنت وتدخل الهندسه الاجتماعيه النفسيه	3,87

Comme pour l'Anglais, nous avons généré dix paraphrases pour chaque réponse modèle du datastet « CCASAG », ce qui nous a donné un ensemble de 480 paraphrases générées au total.

3-2 Métriques d'évaluation de l'ASAG avec Générateur de Paraphrases

Dans le but de mesurer la qualité d'un système plusieurs métriques peuvent être utilisées, dans notre travail, nous utilisons la corrélation selon le coefficient de Pearson (r) et l'erreur moyenne quadratique (RMSE), ces deux mesures sont celles utilisés dans tous les travaux d'évaluation

automatique cités précédemment. Nous définissons ces deux métriques en ce qui suit.

3-2-1 Le coefficient de Pearson [97], [98]

Ce coefficient détermine l'intensité et le sens de la linéarité entre deux variables quantitatives continues. Cette valeur est comprise entre 1 et -1, elle est interprétée de la façon suivante :

- Entre -1 et -0.5 : corrélation forte négative.
- Supérieure à -0.5 et inférieure à 0 : Corrélation faible négative.
- Entre 0.5 et 1 : corrélation forte positive.
- Supérieure à 0 et inférieure à 0.5: corrélation faible positive.
- Egale à 0 : l'absence de la linéarité.

Le coefficient de Pearson entre x et y est calculé selon la formule suivante :

$$r = \frac{cov(x,y)}{\sigma_x \, . \, \sigma_y}$$

Où, $cov(x,y)$ représente la covariance entre x et y, $\sigma_x \, . \, \sigma_y$ représentent le produit non nul entre les écarts types de x et y

Dans notre cas nous allons calculer le coefficient de Pearson entre le vecteur x (représentant l'ensemble des notes données par les enseignants) et le vecteur y, contenant les notes générées par l'algorithme d'évaluation automatique.

3-2-2 L'erreur quadratique RMSE (Root Mean Squared Error)

La mesure RMSE est utilisée généralement pour mesurer la différence entre les valeurs d'échantillon (ou de population) prédites par un modèle et les valeurs observées. L'objectif est de minimiser cette valeur [99].

Elle est calculée selon la formule suivante :

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)}{n}}$$

Où l'ensemble des y représente les valeurs observées, les ŷ représentent les valeurs prédites par le modèle et n est le nombre d'observations.

Dans le cadre de notre travail, nous calculons la valeur RMSE entre le vecteur x (représentant

l'ensemble des notes données par les enseignants) et le vecteur y, contenant les notes générées par l'algorithme d'évaluation automatique.

3-3 Evaluateurs automatiques

Nous avons intégré notre proposition dans le modèle ASAG-V1 pour les deux langues (arabe et anglais), ainsi que dans l'ASAG-V2 pour la langue arabe.

Rappelons que le modèle ASAG-V1 utilise les approches statistiques basées sur le corpus, alors que ASAG-V2 se base sur la régression, une des techniques du Machine Learning.

4- RESULTATS ET DISCUSSIONS

Les résultats des tests réalisés sur ces différents modèles de l'ASAG nous permettent d'estimer les performances du système avec ou sans paraphrases.

Nous exposons dans les deux sous sections suivantes les résultats obtenus pour les 2 modèles V1 et V2.

4-1 Résultats obtenus par ASAG V1(BaseLine).

L'ASAG corrige automatiquement les réponses des étudiants en les comparants aux différentes paraphrases générées par EDAM.

Pour chaque réponse de l'étudiant 10 scores ont été renvoyés, nous avons retenu le meilleur. Après évaluation des notes obtenues par rapport aux notes manuelles avec les métriques Coefficients de Pearson et RMSE, nous les comparons avec ceux renvoyés par les mêmes modèles sans paraphrases.

Nous présentons les résultats obtenus par les deux métriques d'évaluation pour les deux jeux d'essai Anglais et Arabe dans les deux tableaux suivants.

Tableau IV-4 Résultats obtenu pour le Dataset Anglais de Mohler - WE = FastText

Stem	Indicateur	ASAG-V1	
		Sans paraphrases	Avec Paraphrases
Sans stem	**Corrélation**	0,4238	**0,6079**
	RMSE	1,4627	**1,2250**
Porter Stem	**Corrélation**	0,4212	**0,6343**
	RMSE	1,3869	**1,1667**
Snow stem	**Corrélation**	0,4230	**0,6321**
	RMSE	1,3800	**1,1572**

Le tableau IV-4 comporte les résultats obtenus par le coefficient de corrélation de Pearson calculé entre la note obtenue par l'évaluateur automatique, et la moyenne des annotations manuelles, ainsi que l'erreur quadratique moyenne (RMSE).

Les auteurs de l'évaluateur ont proposé différents modèles selon l'utilisation du stemmer ou non. Ils ont calculé les résultats après pondération des fréquences de mots et prise en charge de l'analyseur morphosyntaxique (POS) pour chacun des modèles proposés (détail des fréquences TFminMAX, POS et choix des Stemmers en Annexe B).

Nous avons repris le travail des auteurs avec tous les paramètres utilisés, nous l'avons appliqué pour toutes les paraphrases générées.

Le taux de corrélation obtenu sans paraphrases varie entre 42,12% et 42,38%, ce taux est moyennement faible, presque 60% des notes obtenues sont très loin des notes manuelles.

Quant à l'erreur moyenne quadratique (RMSE), lors de l'évaluation sans paraphrases, les auteurs ont minimisé l'erreur de 1,4627 jusqu'à 1,38 en appliquant le stemmer SNOW STEM.

Notre proposition, basée sur l'intégration des paraphrases, a donné de meilleurs résultats, la corrélation s'est nettement améliorée, elle est passée de 42.38% (meilleur taux sans paraphrase) à 63,43%, d'autant le taux obtenu est très proche de la corrélation entre les deux annotateurs humains, ce dernier était de 64.43% selon le coefficient de Pearson.

De même, la dispersion entre les notes obtenues automatiquement et les notes manuelles a largement diminué. La RMSE calculée est passée de 1.38 (plus faible taux donné sans paraphrases) à 1.1572 (plus faible taux donné avec les paraphrases).

Ces résultats nous permettent de conclure que notre proposition est validée par ce premier travail pour la langue anglaise.

Nous illustrons dans le tableau IV-5, Les résultats obtenus avec cet évaluateur pour la langue arabe.

Tableau IV-5 Les résultats obtenus avec les évaluateurs sans régression pour la langue arabe

Stem	Indicateur	ASAG-V1	
		Sans paraphrases	Avec Paraphrases
Sans stem	**Correlation**	0,6049	**0,6677**
	RMSE	1,8483	**1,4420**
Tasha Light Stem	**Correlation**	0,5863	**0,6786**
	RMSE	1,7740	**1,3850**
Tasha Lourd stem	**Correlation**	0,5737	**0,6669**
	RMSE	1,7778	**1,3678**
ISRI stem	**Correlation**	0,5646	**0,6776**
	RMSE	1,7837	**1,3509**

Pour la langue Arabe, quatre modèles ont été déployés selon le choix du stemmer. Les résultats ont été calculé de la même manière que pour les modèles anglais (La pondération TFminMAX, POS).

Selon le tableau IV-5, nous pouvons dire que les résultats pour l'arabe étaient nettement mieux que pour l'Anglais. La corrélation obtenue sans paraphrases a varié entre 56,46% et 60,49%, avec paraphrase entre 66,69% et 67,86%.

De même, l'intégration des paraphrases a diminué le taux d'erreur RMSE, ce dernier qui était entre 1,8483 et 1,7740, a diminué pour être dans l'intervalle [1,4420, 1.3509].

Les résultats obtenus démontrent que l'intégration de notre générateur EDAM a nettement amélioré les résultats pour les deux langues Arabe et Anglais. Cependant, on peut dire que l'entrainement effectué en langue arabe aurait pu augmenter la précision de l'évaluateur automatique, si le dataset d'entrainement était plus riche et de meilleure qualité.

4-2 Résultats obtenus après intégration avec l'ASAG V2

En intégrant notre générateur EDAM à ASAG V2., nous avons obtenu les résultats pour l'Arabe

illustrés dans le tableau IV-6.

Tableau IV-6 Les résultats obtenus avec l'ASAG V2 pour la langue Arabe

Métriques d'évaluation	Sans Paraphrases	Avec paraphrases
Corrélation	**0,7794**	**0,8892**
RMSE	**0,8968**	**0,6955**

Nous relevons dans le tableau IV-7 le taux de corrélation selon le coefficient de Pearson, ainsi que le taux de dispersion calculé par RMSE pour le modèle avec régression de [r=0,7794 & RMSE = 0.8968].

Le tableau englobe les résultats pour ces deux métriques renvoyées, avec et sans paraphrases pour les deux jeux d'essais dédiés à l'Arabe.

Ce modèle a renvoyé de très bons résultats que ce soit avec ou sans paraphrases, (88,92% et 77,94% de corrélation, 0,6955 et 0,8968 d'écarts RMSE respectivement).

Nous confirmons que l'intégration de paraphrases a impacté positivement les résultats, le taux de corrélation a augmenté de 10.98%, de plus, l'erreur a diminué de 0,8968 à 0,6955.

4-3 Outil graphique pour la génération de paraphrase et Evaluation automatique

Pour une meilleure visibilité de notre travail et des résultats obtenus, nous avons développé un outil graphique. Cet outil permet d'exploiter l'outil d'évaluation ainsi que le générateur de paraphrase indépendamment de la plateforme Moodle. Ainsi notre outil peut être utilisé sur tout autre système ou application Web et encore de bureau. Elle permet aussi à l'enseignant au besoin de valider lui-même les paraphrases de sa réponse modèle.

Dans ce qui suit, nous illustrons quelques captures d'écran des interfaces graphiques du notre outil.

La première interface illustrée dans la Figure IV.2 montre notre première foctionnalité portant sur la génération des paraphrases.

Figure IV-2 Interface de génération des paraphrases

Cette interface permet à l'utilisateur de générer les paraphrases relatives à la phrase saisie en arabe ou en anglais, ainsi que les scores BLEU et GLEU associées à ces dernières. Elle lui permet aussi de filtrer les résultats selon un seuil de BLEU ou GLEU à définir.

L'interface suivant permettant le calcul du score automatique est présentée dans la Figure IV.3.

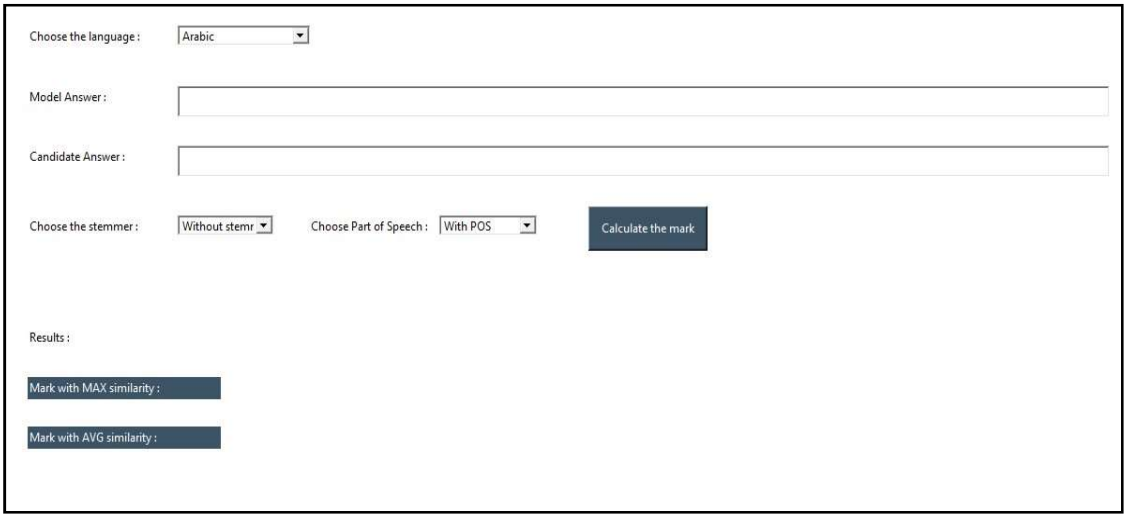


Figure IV-3 Interface de l'évaluation automatique

Notre application permet aussi de calculer le score entre une réponse modèle et une réponse candidate. Cette évaluation automatique intègre le module de génération de paraphrases EDAM. L'utilisateur aura la possibilité de spécifier un certain nombre de paramètres (Stem, POS).

La figure IV.4 montre notre troisième interface, permettant de générer des corpus de

paraphrases.

Figure IV-4 Interface de génération des corpus de paraphrases

Cette rubrique offre la possibilité de générer un corpus de paraphrases sous format de fichier texte à partir d'un fichier contenant les phrases sources.

5- SYNTHESE

L'étude expérimentale effectuée nous a permis d'évaluer de manière extrinsèque le générateur de paraphrase en l'intégrant à l'Outil ASAG. Nous aboutissons aux conclusions suivantes :

- Le taux de corrélation obtenu lors des différents tests démontre que la comparaison des réponses des étudiants à plusieurs réponses modèles donne une meilleure précision que de les comparer avec une seule réponse modèle.
- La dispersion entre les notes obtenues automatiques et les notes références (des annotateurs) a largement diminuée ce qui démontre la qualité de la prédiction s'est nettement améliorée.

Ainsi la génération de paraphrases a un impact important à l'amélioration de la performance des outils d'évaluation automatiques. Les tests conduits dans les deux langues l'arabe et l'anglais nous permet de valider la généricité de nos modèles et leur indépendance vis-à-vis de la langue.

6- CONCLUSION

Dans ce chapitre nous avons intégré notre générateur de paraphrases EDAM au sein du système d'évaluation automatique de réponses courtes (ASAG) pour la langue Arabe.

Cette intégration nous a permis de valider notre proposition en améliorant considérablement les performances du système et en minimisant la dispersion des notes obtenues automatiquement et celles données manuellement.

Nous avons constaté lors de nos tests sur les différents évaluateurs automatiques que l'Intelligence Artificielle a amélioré nettement les performances des systèmes ASAGs.

La corrélation entre la moyenne des notes manuelles et les notes de la correction automatique dépasse la corrélation entre les deux annotateurs arabes (88,92%, 83% Respectivement). Ceci signifie que la correction automatique peut s'avérer plus objective et plus précise que les corrections manuelles.

CONCLUSION GENERALE

Nous nous sommes intéressés dans ce travail à la génération automatique des paraphrases dans le but d'améliorer la précision de notation des systèmes d'évaluation automatique de réponses courtes (ASAG).

Notre étude était concentrée sur l'application du nouveau paradigme de l'Intelligence Artificielle, à savoir le Deep Learning, dans cette tâche du TAL à savoir la génération de paraphrases.

Nous avons commencé par présenter les concepts fondamentaux concepts du Deep Learning en se basant essentiellement sur les différentes techniques et modèles « séquence à séquence » appropriés à la nature des entrées et sorties liées à notre problématique et objectif (des phrases sources et des paraphrases générées).

Dans l'état de l'art, nous avons fait une synthèse des approches de génération des paraphrases existantes et les travaux connexes les plus récents. D'après la synthèse effectuée, nous avons conclu que les travaux les plus prometteurs ont intégré des techniques de Deep Learning. Ils ont utilisé des approches basées sur l'exploitation des ressources linguistiques. La majorité de ces travaux ne traitent pas la langue Arabe. En ce sens nous considérons que notre travail est à notre connaissance le premier à explorer les techniques du deep learning pour la langue arabe dans le contexte de la génération de paraphrase.

A partir de cette recherche bibliographique, nous avons choisi de développer le modèle Bi-LSTM comme une solution de base à laquelle nous pouvons nous référer pour apprécier l'amélioration avec un autre modèle. Nous avons déployé par la suite une des plus récentes techniques du Deep Learning à savoir l'encodeur/décodeur (ED). Nous avons enrichi ce modèle en lui incorporant le Mécanisme d'Attention, nous avons nommé cette approche « EDAM ».

Les trois modèles que nous avons proposés sont basées sur l'exploitation des ressources linguistiques ; un choix justifié dans la littérature. En effet nos modèles sont fondés sur des corpus monolingues parallèles permettant de les alimenter lors de la phase d'entrainement.

Notre travail est dédié à la langue arabe, cependant, nous avons choisi de tester nos approches pour l'anglais et l'arabe. Ceci nous a permis de valider nos solutions et de comparer nos résultats avec ceux des autres travaux de la littérature.

Nous avons effectué une première **évaluation intrinsèque** des modèles de génération de la paraphrase. A cet effet, l'évaluation **quantitative** a permis de montrer que notre modèle EDAM, qui peut générer jusqu'à 10 paraphrases, présente des résultats comparables avec certains travaux et surpasse d'autres travaux récents de la littérature. Dans toute tâche du TAL, l'expertise humaine est plus que nécessaire. Dans ce sens nous avons fait appel à des experts humains en anglais et en arabe pour valider notre générateur sur l'aspect « relevance » et « readability » incluant les aspects sémantiques et de forme (syntaxe). Le résultat de cette expertise humaine, réalisée sur un échantillon de 100 phrases pris aléatoirement du jeu de test, confirme bien la qualité très satisfaisante du générateur syntaxiquement et sémantiquement.

L'évaluation **extrinsèque** du générateur a été réalisée par rapport à la tâche de l'évaluation automatique des réponses courtes. En effet, nous avons intégré notre générateur « EDAM » au sein de deux versions de systèmes d'évaluation automatiques de réponses courtes. Le générateur permet à partir d'une réponse modèle de l'enseignant de générer plusieurs réponses paraphrases portant le même sens. L'ASAG compare alors la réponse de l'étudiant de manière plus « juste » non pas à une réponse modèle unique mais à plusieurs réponses modèles incluant toutes les variations linguistiques possibles grâce à EDAM. Ainsi, le générateur de paraphrase décharge l'enseignant de la tâche fastidieuse d'imaginer et de construire manuellement toutes les formulations possibles de la réponse modèle.

L'intégration des paraphrases dans la correction automatique a nettement amélioré les deux systèmes avec toutes les variantes appliquées.

Ainsi la génération de paraphrases a un impact important à l'amélioration de la performance des outils d'évaluation automatiques. Les tests conduits dans les deux langues l'arabe et l'anglais nous permettent de valider la généricité de nos modèles et leur indépendance vis-à-vis de la langue.

Malgré les résultats très prometteurs, quelques limites ont été soulevées à savoir :

- Le générateur a été entrainé sur des petits datasets, ceci est un grand inconvénient dans le Deep Learning ;
- Le datastet arabe d'entrainement n'est pas riche en contenu, il ne traite pas beaucoup de

domaines, d'autant plus il comporte trop d'erreurs grammaticales et syntaxiques. Le manque de ressources en langue arabe persiste encore comme un défi qu'il faut sérieusement considérer.

Afin de remédier à ces limites et améliorer la qualité du générateur, nous proposerons en perspectives :

- ✓ Entrainer le modèle EDAM sur des dataset plus volumineux avec une variété de domaines.
- ✓ Entrainer les modèles à base de Deep Learning sur des serveurs plus puissants avec des plus d'époques.
- ✓ Proposer les nouvelles techniques de Deep Learning tel que les réseaux adverses génératifs (GAN) dans la génération de paraphrases et en explorer l'impact.

RÉFÉRENCES BIBLIOGRAPHIQUES

[1] S. J. Russell and P. Norvig, “Artificial Intelligence-A Modern Approach (3rd internat. edn.).” Pearson Education, 2010.

[2] E. Heer, *The age of intelligent machines*, vol. 28, no. 1. MIT press Cambridge, 1993.

[3] F. S. Osorio, “Inss : Un Systeme Hybride Neuro-Symbolique Pour L’Apprentissage Automatique Constructif.” Institut National Polytechnique de Grenoble-INPG, p. 315, 2004.

[4] H. A. Simon, “Why Should Machines Learn?,” in *Machine Learning*, Elsevier, 1983, pp. 25–37.

[5] “Qu’est-ce que L’intelligence artificielle (IA) ? | Le Data Scientist.” [Online]. Available: https://ledatascientist.com/l-intelligence-artificielle-ia/?fbclid=IwAR3o8FZXABtT4m1ig63ri-P3J4XK5RHw2ius27PsuhZIvCRmKQXB6PWaY1Q. [Accessed: 25-Aug-2020].

[6] Y. Y. Chen, Y. H. Lin, C. C. Kung, M. H. Chung, and I. H. Yen, “Design and implementation of cloud analytics-assisted smart power meters considering advanced artificial intelligence as edge analytics in demand-side management for smart homes,” *Sensors (Switzerland)*, vol. 19, no. 9, p. 2047, 2019, doi: 10.3390/s19092047.

[7] P. M. Rowiński, A. Piotrowski, and J. J. Napiórkowski, “Les techniques de réseaux de neurones artificiels sont-elles pertinentes pour estimer le coefficient de dispersion longitudinale en rivières?,” *Hydrol. Sci. J.*, vol. 50, no. 1, pp. 175–187, 2005, doi: 10.1623/hysj.50.1.175.56339.

[8] “Deep Learning in a nutshell,” 01-May-2016. [Online]. Available: https://www.slideshare.net/chinhuichen/20160226-deep-learning-in-a-nutshell. [Accessed: 25-Aug-2020].

[9] “Activation Functions | Fundamentals Of Deep Learning,” 30-Jan-2020. [Online]. Available: https://www.analyticsvidhya.com/blog/2020/01/fundamentals-deep-learning-activation-functions-when-to-use-them/. [Accessed: 26-Aug-2020].

[10] “Activation Functions — ML Glossary documentation.” [Online]. Available: https://ml-cheatsheet.readthedocs.io/en/latest/activation_functions.html. [Accessed: 26-Aug-2020].

[11] J. Schmidhuber, “Deep Learning in neural networks: An overview,” *Neural Networks*, vol. 61, pp. 85–117, 2015, doi: 10.1016/j.neunet.2014.09.003.

[12] Y. Bengio, A. Courville, and P. Vincent, “Representation learning: A review and new perspectives,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, 2013, doi: 10.1109/TPAMI.2013.50.

[13] D. E. Rumelhart and J. L. McClelland, “Schemata and Sequential Thought Processes in PDP Models,” *Parallel Distrib. Process.*, vol. 2, pp. 3–57, 2020, doi: 10.7551/mitpress/5236.003.0004.

[14] M.Nayak, “Introduction to the Architecture of Recurrent Neural Networks (RNNs) | by Manish Nayak | Towards AI—Multidisciplinary Science Journal | Medium,” 29-Jun-2019. [Online]. Available: https://medium.com/towards-artificial-intelligence/introduction-to-the-architecture-of-recurrent-neural-networks-rnns-a277007984b7. [Accessed: 25-Aug-2020].

[15] A.Ng, K.Katanforoosh, and Y.Bensouda, “Bidirectional RNN - Recurrent Neural Networks | Coursera.” [Online]. Available: https://www.coursera.org/lecture/nlp-sequence-models/bidirectional-rnn-fyXnn. [Accessed: 25-Aug-2020].

[16] A. C. I.Goodfellow, Y.Bengio, *Deep learning*, vol. 29, no. 7553. MIT press Massachusetts, USA:, 2019.

[17] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[18] “Understanding LSTM Networks -- colah’s blog,” 27-Aug-2015. [Online]. Available: https://colah.github.io/posts/2015-08-Understanding-LSTMs/. [Accessed: 25-Aug-2020].

[19] A.Raghav, “Bi-LSTM. What is a neural network? Just like our… | by Raghav Aggarwal | Medium,” 2019. [Online]. Available: https://medium.com/@raghavaggarwal0089/bi-lstm-bc3d68da8bd0. [Accessed: 25-Aug-2020].

[20] I. Sutskever, O. Vinyals, and Q. V. Le, “Sequence to sequence learning with neural networks,” in *Advances in Neural Information Processing Systems*, 2014, vol. 4, no. January, pp. 3104–3112.

[21] E.Culurciello, “Sequence-to-sequence neural networks | by Eugenio Culurciello | Medium,” 16-Apr-2019. [Online]. Available: https://medium.com/@culurciello/sequence-to-sequence-neural-networks-3d27e72290fe. [Accessed: 25-Aug-2020].

[22] Q.Wu, “A Brief Overview of Attention Mechanism | by Synced | SyncedReview | Medium,” 2017. [Online]. Available: https://medium.com/syncedreview/a-brief-overview-of-attention-mechanism-13c578ba9129. [Accessed: 25-Aug-2020].

[23] C. Raffel and D. P. W. Ellis, “Feed-Forward Networks with Attention Can Solve Some Long-Term Memory Problems,” Dec. 2015.

[24] A. R. Martinez, “Natural language processing,” *Wiley Interdiscip. Rev. Comput. Stat.*, vol. 2, no. 3, pp. 352–357, 2010, doi: 10.1002/wics.76.

[25] T. Young, D. Hazarika, S. Poria, and E. Cambria, “Recent trends in deep learning based natural language processing [Review Article],” *IEEE Comput. Intell. Mag.*, vol. 13, no. 3, pp. 55–75, 2018, doi: 10.1109/MCI.2018.2840738.

[26] N. Madnani and B. J. Dorr, “Generating phrasal and sentential paraphrases: A survey of data-driven methods,” *Comput. Linguist.*, vol. 36, no. 3, pp. 341–387, 2010, doi: 10.1162/coli_a_00002.

[27] S. Meshram, “Review on NLP Paraphrase Detection Approaches,” vol. 4, no. 11, pp. 351–354, 2019.

[28] H. Bouamor, “Etude De La Paraphrase Sous-Phrastique En Traitement Automatique Des Langues,” p. 180, 2012.

[29] Z. S. Harris, “Distributional Structure,” WORD, vol. 10, no. 2–3, pp. 146–162, 1954, doi: 10.1080/00437956.1954.11659520.

[30] D. Lin and P. Pantel, “Discovery of inference rules for question-,” *Nat. Lang. Eng.*, vol. 7, no. 4, pp. 343–360, 2001, doi: 10.1017/S1351324901002765.

[31] M. Paşca and P. Dienes, “Aligning needles in a haystack: Paraphrase acquisition across the Web,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 3651 LNAI, pp. 119–130, 2005, doi: 10.1007/11562214_11.

[32] P. Pantel *et al.*, “2007NAACLlearning inferential selectional preference.pdf,” no. April, pp. 564–571, 2007.

[33] S. Fernando and M. Stevenson, “A Semantic Similarity Approach to Paraphrase Detection,” *Proc. 11th Annu. Res. Colloq. UK Spec. Interes. Gr. Comput. Linguist. (CLUK 2008)*, pp. 45–52, 2008, doi: 10.1.1.144.4680.

[34] R. Bhagat and D. Ravichandran, “Large scale acquisition of paraphrases for learning surface patterns,” *ACL-08 HLT - 46th Annu. Meet. Assoc. Comput. Linguist. Hum. Lang. Technol. Proc. Conf.*, no. June, pp. 674–682, 2008.

[35] Y. Shinyama, S. Sekine, and K. Sudo, “Automatic paraphrase acquisition from news articles,” in *Proceedings of the second international conference on Human Language Technology Research*, 2002, pp. 313–318, doi: 10.3115/1289189.1289218.

[36] S. Sekine, “Extracting synonymous expressions from multiple newspaper documents,” in *Proceedings of the ANLP-2001 Workshop on Automated Paraphrasing*, 2001.

[37] B. Dolan, C. Quirk, and C. Brockett, “Unsupervised construction of large paraphrase corpora,” in *Proceedings of the 20th international conference on Computational Linguistics*, 2004, pp. 350-es, doi: 10.3115/1220355.1220406.

[38] R. Barzilay and L. Lee, “Learning to Paraphrase: An Unsupervised Approach Using Multiple-Sequence Alignment,” in *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, 2003, pp. 16–23.

[39] R. Barzilay and K. R. McKeown, “Extracting paraphrases from a parallel corpus,” in *Proceedings of the 39th annual meeting of the Association for Computational Linguistics*, 2001, pp. 50–57, doi: 10.3115/1073012.1073020.

[40] D. Bernhard and I. Gurevych, “Answering learners’ questions by retrieving question paraphrases from social Q&A sites,” in *Proceedings of the third workshop on innovative use of NLP for building educational applications*, 2008, pp. 44–52, doi: 10.3115/1631836.1631842.

[41] G. M. Emelyanov, D. V. Mikhailov, and A. P. Kozlov, “Relevance of a Set of Topical Texts to a Knowledge Unit and the Estimation of the Closeness of Linguistic Forms of Its Expression to a Semantic Pattern,” *Pattern Recognit. Image Anal.*, vol. 28, no. 4, pp. 771–782, 2018, doi: 10.1134/S1054661818040090.

[42] K. Sarkar, “KS-JU@DPIL-FIRE2016: Detecting paraphrases in Indian languages using multinomial logistic regression model,” *CEUR Workshop Proc.*, vol. 1737, pp. 250–255, 2016.

[43] C. Bannard and C. Callison-Burch, “Paraphrasing with bilingual parallel corpora,” in *ACL-05 - 43rd Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, 2005, pp. 597–604, doi: 10.3115/1219840.1219914.

[44] A. Patry and P. Langlais, “Identifying Parallel Documents from a Large Bilingual Collection of Texts: Application to Parallel Article Extraction in Wikipedia,” in *ACL WS Building and Using Comparable Corpora: Comparable Corpora and the Web*, 2011, no. June, pp. 87–95.

[45] F. Al-Raisi, W. Lin, and A. Bourai, “A Monolingual Parallel Corpus of Arabic,” *Procedia Comput. Sci.*, vol. 142, pp. 334–338, 2018, doi: 10.1016/j.procs.2018.10.487.

[46] P. Koehn, “Europarl : A Parallel Corpus for Statistical Machine Translation,” in *MT Summit*, 2005, vol. 11, pp. 79--86, doi: 10.3115/1626355.1626380.

[47] R. Socher, E. H. Huang, J. Pennington, A. Y. Ng, and C. D. Manning, “Dynamic pooling and unfolding recursive autoencoders for paraphrase detection,” in *Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011, NIPS 2011*, 2011, pp. 801–809.

[48] W. Yin and H. Schütze, “Convolutional neural network for paraphrase identification,” in *NAACL HLT 2015 - 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference*, 2015, pp. 901–911, doi: 10.3115/v1/n15-1091.

[49] D. Zeng, H. Zhang, L. Xiang, J. Wang, and G. Ji, “User-Oriented Paraphrase Generation with Keywords Controlled Network,” *IEEE Access*, vol. 7, pp. 80542–80551, 2019, doi: 10.1109/ACCESS.2019.2923057.

[50] A. Prakash *et al.*, “Neural paraphrase generation with stacked residual LSTM Networks,” *COLING 2016 - 26th Int. Conf. Comput. Linguist. Proc. COLING 2016 Tech. Pap.*, pp. 2923–2934, 2016.

[51] S. Huang, Y. Wu, F. Wei, and Z. Luan, “Dictionary-Guided Editing Networks for Paraphrase Generation,” *Proc. AAAI Conf. Artif. Intell.*, vol. 33, pp. 6546–6553, 2019, doi: 10.1609/aaai.v33i01.33016546.

[52] A. M. Ben Omran and M. J. Ab Aziz, “Automatic essay grading system for short answers in English language,” *J. Comput. Sci.*, vol. 9, no. 10, pp. 1369–1382, 2013, doi: 10.3844/jcssp.2013.1369.1382.

[53] N. Nakatsu, Y. Kambayashi, and S. Yajima, “A longest common subsequence algorithm suitable for similar text strings,” *Acta Inform.*, vol. 18, no. 2, pp. 171–179, 1982.

[54] S.Abassi and H.Boulhouache, “Université Saad Dahlab blida 1: Génération automatique de corpus de réponses modèles dans un système d’évaluation automatique des réponses courtes,” 29-Sep-2019. [Online]. Available: http://di.univ-blida.dz:8080/jspui/handle/123456789/3599. [Accessed: 25-Aug-2020].

[55] E. M. B. Nagoudi and D. Schwab, “Semantic Similarity of Arabic Sentences with Word Embeddings,” 2017, pp. 18–24, doi: 10.18653/v1/w17-1303.

[56] A. Islam and D. Inkpen, “Semantic Text Similarity Using Corpus-Based Word Similarity and String Similarity,” *ACM Trans. Knowl. Discov. Data*, vol. 2, no. 2, pp. 1–25, 2008, doi: 10.1145/1376815.1376819.

[57] J. Macqueen, “Some methods for classification and analysis,” in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, 1967, vol. 233, no. 233, pp. 281–297.

[58] K. He, X. Zhang, S. Ren, and J. Sun, “Identity mappings in deep residual networks,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2016, vol. 9908 LNCS, pp. 630–645, doi: 10.1007/978-3-319-46493-0_38.

[59] K. P. Nelson, “Reduced Perplexity: A simplified perspective on assessing probabilistic forecasts,” *arXiv Prepr. arXiv1603.08830*, 2016.

[60] K. Papineni, S. Roukos, T. Ward, W. Zhu, and Y. Heights, “IBM Research Report Bleu : a Method for Automatic Evaluation of Machine Translation,” *Science (80-.).*, vol. 22176, pp. 1–10, 2001, doi: 10.3115/1073083.1073135.

[61] A. Lavie and A. Agarwal, “METEOR: An automatic metric for MT evaluation with high levels of correlation with human judgments,” *Proc. Second Work. Stat. Mach. Transl.*, vol. 0, no. June, pp. 228–231, 2007.

[62] M. Snover, B. Dorr, R. Schwartz, L. Micciulla, and J. Makhoul, “A study of translation edit rate with targeted human annotation,” *AMTA 2006 - Proc. 7th Conf. Assoc. Mach. Transl. Am. Visions Futur. Mach. Transl.*, no. August, pp. 223–231, 2006.

[63] E. Loper and S. Bird, “NLTK: The Natural Language Toolkit,” *arXiv Prepr. cs/0205028*, 2002.

[64] J. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, “Attention-based models for speech recognition,” in *Advances in Neural Information Processing Systems*, 2015, vol. 2015-Janua, pp. 577–585.

[65] J. Walker, K. Marino, A. Gupta, and M. Hebert, “The pose knows: Video forecasting by generating pose futures,” in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 3332–3341.

[66] C. Florescu and C. Caragea, “PositionRank: An unsupervised approach to keyphrase extraction from scholarly documents,” in *ACL 2017 - 55th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers)*, 2017, vol. 1, pp. 1105–1115, doi: 10.18653/v1/P17-1102.

[67] M. Alkhatib and K. Shaalan, “Paraphrasing Arabic Metaphor with Neural Machine Translation,” *Procedia Comput. Sci.*, vol. 142, no. ACLing, pp. 308–314, 2018, doi: 10.1016/j.procs.2018.10.493.

[68] D. Bahdanau, K. H. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate,” *3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc.*,

2015.

[69] F. Al-Raisi, A. Bourai, and W. Lin, “Neural Symbolic Arabic Paraphrasing with Automatic Evaluation,” pp. 01–13, 2018, doi: 10.5121/csit.2018.80601.

[70] A. Gupta, A. Agarwal, P. Singh, and P. Rai, “A deep generative framework for paraphrase generation,” *32nd AAAI Conf. Artif. Intell. AAAI 2018*, pp. 5149–5156, 2018.

[71] E. Pavlick, P. Rastogi, J. Ganitkevitch, B. Van Durme, and C. Callison-Burch, “PPDB 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification,” in *ACL-IJCNLP 2015 - 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, Proceedings of the Conference*, 2015, vol. 2, pp. 425–430, doi: 10.3115/v1/p15-2070.

[72] A. Fader, L. Zettlemoyer, and O. Etzioni, “Paraphrase-driven learning for open question answering,” in *ACL 2013 - 51st Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, 2013, vol. 1, pp. 1608–1618.

[73] T. Y. Lin *et al.*, “Microsoft COCO: Common objects in context,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2014, vol. 8693 LNCS, no. PART 5, pp. 740–755, doi: 10.1007/978-3-319-10602-1_48.

[74] S. In, “First Quora Dataset Release: Question Pairs,” *data. quora. com*, pp. 2–3, 2017.

[75] M. Iyyer, J. Wieting, K. Gimpel, and L. Zettlemoyer, “Adversarial Example Generation with Syntactically Controlled Paraphrase Networks,” *arXiv Prepr. arXiv1804.06059*, pp. 1875–1885, 2018, doi: 10.18653/v1/n18-1170.

[76] D. L. Chen and W. B. Dolan, “Collecting highly parallel data for paraphrase evaluation,” in *ACL-HLT 2011 - Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 2011, vol. 1, pp. 190–200.

[77] C. Liu, D. Dahlmeier, and H. T. Ng, “PEM: A paraphrase evaluation metric exploiting parallel texts,” in *EMNLP 2010 - Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, 2010, pp. 923–932.

[78] C. Napoles, K. Sakaguchi, M. Post, and J. Tetreault, “Ground truth for grammatical error correction metrics,” in *ACL-IJCNLP 2015 - 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, Proceedings of the Conference*, 2015, vol. 2, pp. 588–593.

[79] C. Napoles, K. Sakaguchi, M. Post, and J. Tetreault, “GLEU Without Tuning,” May 2016.

[80] D. Klakow and J. Peters, “Testing the correlation of word error rate and perplexity,” *Speech Commun.*, vol. 38, no. 1–2, pp. 19–28, 2002, doi: 10.1016/S0167-6393(01)00041-3.

[81] Z. Li, X. Jiang, L. Shang, and H. Li, “Paraphrase Generation with Deep Reinforcement Learning,” pp. 3865–3878, 2019, doi: 10.18653/v1/d18-1421.

[82] F.Al-Raisi, “Index of /~fraisi/arabic/arparallel,” 23-Sep-2018. [Online]. Available: http://www.cs.cmu.edu/~fraisi/arabic/arparallel/. [Accessed: 25-Aug-2020].

[83] “First Quora Dataset Release: Question Pairs - Data @ Quora - Quora.” [Online]. Available: https://www.quora.com/q/quoradata/First-Quora-Dataset-Release-Question-Pairs. [Accessed: 25-Aug-2020].

[84] D.Karani, “Introduction to Word Embedding and Word2Vec | by Dhruvil Karani | Towards Data Science,” 01-Sep-2018. [Online]. Available: https://towardsdatascience.com/introduction-to-word-embedding-and-word2vec-652d0c2060fa. [Accessed: 25-Aug-2020].

[85] M. A. Zahran, A. Magooda, A. Y. Mahgoub, H. Raafat, M. Rashwan, and A. Atyia, “Word representations in vector space and their applications for Arabic,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2015, vol. 9041, pp. 430–443, doi: 10.1007/978-3-319-18111-0_32.

[86] “NLPL word embeddings repository.” [Online]. Available: http://vectors.nlpl.eu/repository/?fbclid=IwAR3xfVYEYqEVYovgrcW5GwWL57nSeL_mUBzMV4EUNYD61xkX_UYrqi7aT08. [Accessed: 25-Aug-2020].

[87] A. K. Vijayakumar *et al.*, “Diverse Beam Search: Decoding Diverse Solutions from Neural Sequence Models,” *arXiv Prepr. arXiv1610.02424*, 2016.

[88] “Évaluer des modèles | Documentation d’AutoML Translation.” [Online]. Available: https://cloud.google.com/translate/automl/docs/evaluate?hl=fr&fbclid=IwAR3f_J5uf2uW_x-sKlboncA-wbXVMJ4FsPFxnT-tjEljPALWBM9uSp8ZjD0. [Accessed: 25-Aug-2020].

[89] Q. Yang *et al.*, “An end-to-end generative architecture for paraphrase generation,” in *EMNLP-IJCNLP 2019 - 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Proceedings of the Conference*, 2020, pp. 3132–3142, doi: 10.18653/v1/d19-1309.

[90] B. Babych, “Automated MT evaluation metrics and their limitations,” *Tradumàtica Tecnol. la traducció*, no. 12, p. 464, 2014, doi: 10.5565/rev/tradumatica.70.

[91] A.Madani and E.Snoussi, “Université Saad Dahlab blida 1: Développement d’un PLUGIN d’évaluation automatique des réponses courtes pour une plateforme de télé-enseignement,” 2019. [Online]. Available: http://di.univ-blida.dz:8080/jspui/handle/123456789/2930. [Accessed: 25-Aug-2020].

[92] “Moodle - Open-source learning platform | Moodle.org.” [Online]. Available: https://moodle.org/. [Accessed: 25-Aug-2020].

[93] M. Hadjersi, O. Benguergoura, « Le machine Learning pour l’Evaluation Automatique des Réponses Courtes : Application à la Langue Arabe». Mémoire master USDB 1. 2019/2020.

[94] “Rada Mihalcea: Downloads.” [Online]. Available: https://web.eecs.umich.edu/~mihalcea/downloads.html. [Accessed: 25-Aug-2020].

[95] “I.Amar setti” F.Oukina, “Université Saad Dahlab blida 1: Elaboration d’un corpus de test pour un système d’évaluation automatique des réponses courtes,” 09-Sep-2019. [Online]. Available: http://di.univ-blida.dz:8080/jspui/handle/123456789/3494. [Accessed: 29-Aug-2020].

[96] M. Mohler and R. Mihalcea, “Text-to-text semantic similarity for automatic short answer grading,” in *EACL 2009 - 12th Conference of the European Chapter of the Association for Computational Linguistics, Proceedings*, 2009, pp. 567–575, doi: 10.3115/1609067.1609130.

[97] “Pearson Correlation - SPSS Tutorials - LibGuides at Kent State University.” [Online]. Available: https://libguides.library.kent.edu/SPSS/PearsonCorr. [Accessed: 25-Aug-2020].

[98] K. Yeager, “LibGuides: SPSS Tutorials: Pearson Correlation.”

[99] “RMS Error.” [Online]. Available: https://statweb.stanford.edu/~susan/courses/s60/split/node60.html. [Accessed: 25-Aug-2020].

[100] J.D’Souza, “Learning POS Tagging & Chunking in NLP | by Jocelyn D’Souza | GreyAtom | Medium,” 04-Apr-2018. [Online]. Available: https://medium.com/greyatom/learning-pos-tagging-chunking-in-nlp-85f7f811a8cb. [Accessed: 25-Aug-2020].

[101] “Overview - CoreNLP.” [Online]. Available: https://stanfordnlp.github.io/CoreNLP/. [Accessed: 25-Aug-2020].

[102] B. Furlan, V. Batanović, and B. Nikolić, “Semantic similarity of short texts in languages with a deficient natural language processing support,” *Decis. Support Syst.*, vol. 55, no. 3, pp. 710–719, 2013, doi: 10.1016/j.dss.2013.02.002.

ANNEXE A

- **Longest Common Subsequence (LCS)**

Afin de trouver la sous séquence commune la plus longue, la réponse d'étudiant et les réponses modèles vont subir un traitement. Ce traitement consiste à transformer chaque phrase en une seule chaine de caractères (éliminer les blancs), et puis chercher la plus longue succession de lettres communes entre la réponse d'étudiant et chacune des réponses modèles, en prenant la valeur maximale. Ce processus est appelé l'algorithme LCS.

La valeur de similarité est calculée comme suit:

$$sim_{LCS} = \frac{2 * |LCS(s1, s2)|}{|s1| + |s2|}$$

{ $LCS(s1, s2)$ représente la sous séquence la plus longue entre les phrases $s1$ et $s2$.
$|s1|$ représente la longuer de la phrase $s1$
$|s2|$ représente la longueur de la phrase 2 }

- **Common Words (COW)**

Le principe général est de calculer le nombre de mots communs entre chacune des phrases modèles et la réponse d'étudiant. Le résultat qui présente la plus grande valeur sera pris en considération.

La similarité entre deux phrases en appliquant COW est calculée selon la formule suivante:

$$sim_{COW} = \frac{2 * c}{|s1| + |s2|}$$

{c représente le nombre de mots communs entre les phrases $s1$ et $s2$
$|s1|$représente la longuer de la phrase $s1$.
$|s2|$ représente la longueur de la phrase $s2$}.

- **Semantic Distance (SD)**

Etant donné deux phrases s1 et s2, avec: s1 = {W_{11}, W_{12}, …, W_{1m}} et s2 = {W_{21}, W_{22}, …, W_{2n}}.

Le processus de Matching consiste à prendre un mot d'une phrase et de le comparer avec chacun des mots de l'autre phrase en calculant le nombre de caractères communs entre ses deux mots. Cette opération est appliquée pour tous les mots de s1(respectivement s2) avec les mots s2(respectivement s1). On fait cette opération entre la phrase de l'étudiant et chacune des réponses modèles, en prenant à la fin la valeur maximale de la distance sémantique.

La formule de calcul est la suivante:

$$sim_{s_d}(s1, s2) = \left(\frac{\sum_{i=1}^{m} a_i}{m} + \frac{\sum_{j=1}^{n} b_j}{n}\right)/2$$

$$a = max\,(s\,(W_{1i}, W_{21}), s\,(W_{1i}, W_{22}), \ldots, s\,(W_{1i}, W_{2n}))$$

$$b = max\,(s\,(W_{11}, W_{2j}), s\,(W_{12}, W_{2j}), \ldots, s\,(W_{1m}, W_{2j}))$$

$\sum_{i=1}^{m} a_i$: représente la somme des nombres de mots matchés a_i divisée par le nombre de mots de la première phrase m.

$\sum_{j=1}^{n} b_j$: représente la somme des nombres de mots matchés b_j divisée par le nombre de mots de la deuxième phrase n.

La valeur de similarité globale est calculée en combinant les valeurs obtenues lors de l'application des trois mesures de similarité citées ci-dessus après les avoir pondérés. La formule est illustrée ci-dessous:

$$sim\,(s_1, s_2) = \lambda_1 * sim_{LCS}\,(s_1, s_2) + \lambda_2 * sim_{COW}\,(s_1, s_2) + \lambda_3 * sim_{s\_d}\,(s_1, s_2)$$

Où les λ représentent les poids attribués aux trois mesures.

ANNEXE B

Nous nous sommes basés sur des algorithmes développés par d'autres binômes lors des années passées, et de cette année.

Pour le travail de cette année, il est basé sur les notions de Machine Learning, tandis que les travaux antérieurs ont intégré des techniques appliquées à la Recherche d'Information telles le stemming et la pondération.

Dans ce qui suit, nous allons expliquer en détails les différents concepts utilisés par l'un de ces algorithmes de l'année passée.

2-2-1 Algorithme d'évaluation automatique sans régression

Afin d'expliquer le fonctionnement de cet algorithme, nous allons tout d'abord illustrer ses différentes étapes dans la figure qui suit, puis nous mettons l'accent sur les détails.

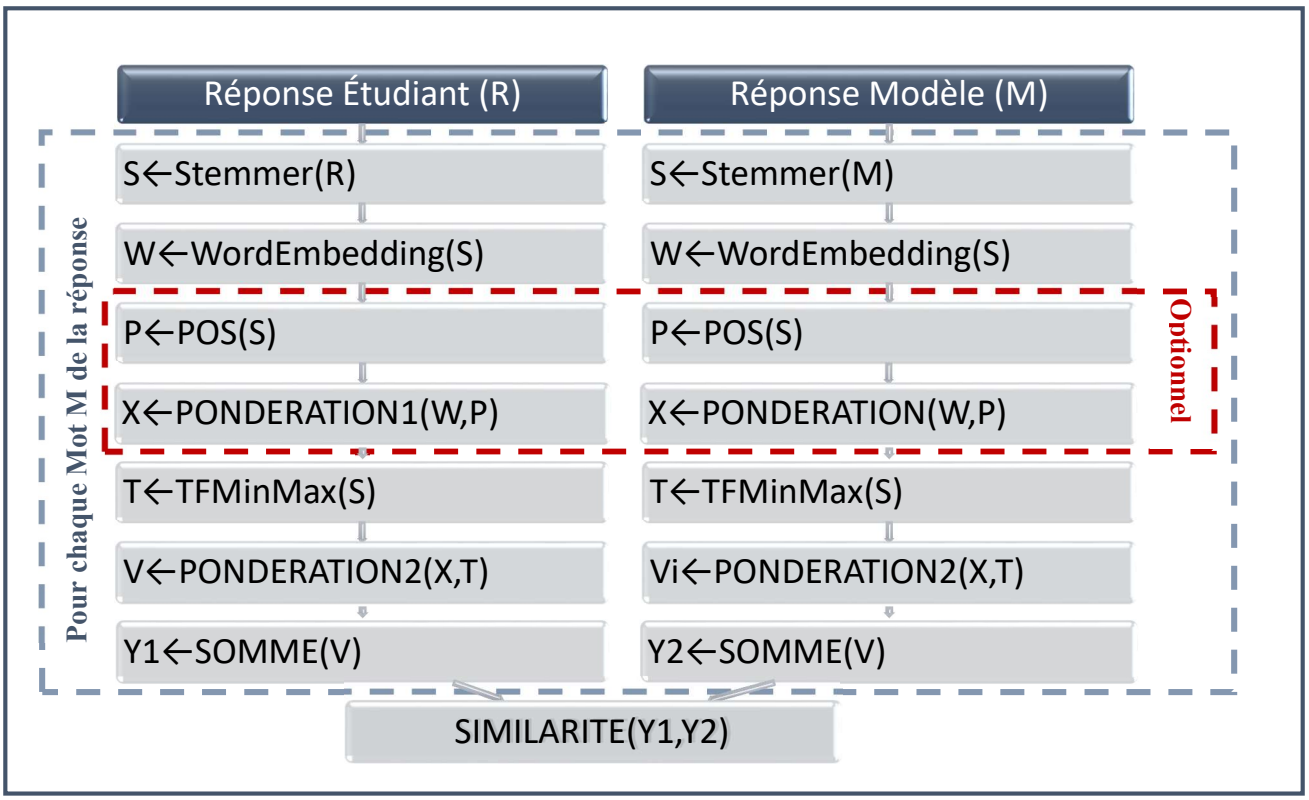


Figure B-1 Algorithme d'évaluation automatique sans régression.

a- L'application de stemmer

Le stemming (racinisation) est le processus de transformer les mots en leurs radicaux, en supprimant le suffixe et/ou le préfixe, et ce, dans le but d'avoir toutes les différentes variations correspondantes à un stem.

L'algorithme d'évaluation cité précédemment utilise 3 différentes techniques de stemming:

- Sans stemmer: aucun stemmer n'est appliquée sur les phrases des enseignants et celles des étudiants.
- Stemmer lourd : consiste à récupérer la racine réelle d'un mot en éliminant les suffixes et préfixes.
- Stemmer léger : cette technique se repose sur la suppression des suffixes et préfixes, sans récupérer la racine réelle du mot.

Au cours de notre utilisation des cet algorithme, nous avons utilisé les stemmers ISRI et Tashaphine (lourd et léger) pour la langue arabe, et les stemmers Snow et Porter pour la langue anglaise.

b- Récupération des vecteurs word embedding

Cette deuxième étape consiste à récupérer les vecteurs relatifs aux mots constituant les réponses des enseignants et des étudiants, après avoir appliqué l'étape A.

Les modèles de word embedding utilisés sont :

- Le word embedding de Zahrane pour la langue arabe.
- Le word embedding FastText pour la langue anglais.

c- Part of speech (POS)

La pondération des mots basée sur l'étiquetage morphosyntaxique, est une technique qui se base sur les caractéristiques syntaxiques et grammaticales de la langue en question.

Un traitement sur le texte est effectué par l'analyseur morphosyntaxique, dans le but de classer les mots dans les différentes classes (Verbe, Nom, etc.). Puis chaque mot sera pondéré selon la valeur attribuée à sa classe, et ce, selon la langue utilisée et ses caractéristiques.

Les vecteurs obtenus de l'étape B seront pondérés selon les poids relatifs aux différents mots,

après avoir effectué l'analyse sur le texte via l'analyseur morphosyntaxique.

Les analyseurs morphosyntaxiques utilisés dans notre cas, sont les suivants :

- Stanford coreTAL pour la langue arabe.
- L'analyseur de nltk pour la langue anglaise.

d- La pondération TF_Min_Max

Afin de calculer cette pondération, nous devons tout d'abord calculer le TFlog pour chaque mot via la formule suivante :

$$TFlog(W) = -log(\frac{cpt}{N})$$

Avec cpt et N qui représentent le nombre d'apparition de mot (W) dans le corpus, et le nombre de mots dans le corpus respectivement.

Les TFlogs obtenus précédemment vont subir une normalisation, et ce, pour obtenir les valeurs finales de TF_Min_Max, en appliquant la formule suivante.

$$TF\_Min\_Max(W) = \frac{TFlog(W)}{Max(TFlogs)}$$

Où le Max (TFlogs), représente la valeur maximale des TFlogs obtenus.

Nous appliquons les pondération TF_Min_Max calculéessur les vecteurs obtenus de l'étape précédente.

e- Somme des vecteurs

Cette technique consiste à faire la somme des vecteurs représentatifs de tous les mots de la phrase, afin d'obtenir un seul vecteur représentant la phrase en question.

Nous appliquons cette technique sur les vecteurs résultants de l'étape D, et ce, pour les deux types des réponses (les réponses des étudiants et les réponses des enseignants).

Nous illustrons ce concept dans la figure suivante, sur un exemple d'une phrase contenant 3 mots.

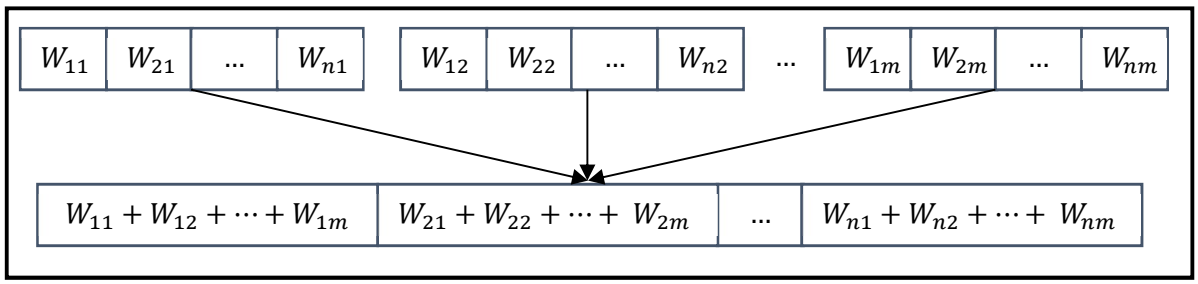


Figure B-2 Somme des vecteurs WordEmbedding des mots d'une réponse.

Cette étape a pour but la préparation des vecteurs pour l'étape prochaine, où nous allons calculer la similarité cosinus entre les vecteurs.

f- Calcul de la similarité et génération du score

Une fois l'étape E est effectuée, nous nous pouvons passer au calcul de la similarité cosinus entre les vecteurs obtenus, relatifs à la réponse de l'enseignant et celle de l'étudiant.

La valeur obtenue sera comprise entre 0 et 1 (0 pour une similarité basse, et 1 pour une similarité haute).

Nous pouvons maintenant générer le score, en multipliant la valeur de similarité par 5, et ce, pour obtenir une note entre 0 et 5.

Dans notre travail, nous allons intégrer le générateur EDAM au sein des deux versions du système, nous comparons par la suite les résultats obtenus.