Résumé de l'article :
Neural Text Generation from Structured Data with Application to the Biography Domain

KHOULADI Salma **LU Xiaohua**

1 Introduction

L'article [1] présente un modèle de neurones pour la génération de texte à partir de concept qui génère la première phrase biographique à partir des informations du 'infobox' en utilisant une large dataset contenant des biographies provenant de Wikipédia.

Keywords: Génération de texte, Modélisation du Langage Naturel (NLM).

Frederick Parker-Rhodes

Born	21 November 1914 Newington, Yorkshire
Died	2 March 1987 (aged 72)
Residence	UK
Nationality	British
Fields	Mycology, Plant Pathology, Mathematics, Linguistics, Computer Science
Known for	Contributions to computational linguistics, combinatorial physics, bit-string physics, plant pathology, and mycology
Author abbrev. (botany)	Park.-Rhodes

figure 1 - Cas d'usage: Boîte d'information Wikipédia de Frederick Parker-Rhodes. "Frederick Parker-Rhodes (21 March 1914 – 21 November 1987) was an English linguist, plant pathologist, computer scientist, mathematician, mystic, and mycologist.".

2 Résumé de l'article

2.1 Travaux associés

Les systèmes de génération traditionnels reposaient sur des règles et des spécifications manuelles. La génération est divisée en trois parties : **Planification du contenu**, définit les parties des champs d'entrée qui doivent être sélectionnées, **Planification des phrases**, détermine les champs sélectionnés qui doivent être traités dans chaque phrase de sortie et **Réalisation**, génère ces phrases. Des approches ultérieures ont combiné deux ou plusieurs de ces parties. L'approche présentée s'inspire des modèles linguistiques utilisés pour le sous-titrage des images, la traduction automatique, la modélisation des conversations et des dialogues. Le modèle discuté utilise un réseau de neurones de type encodeur-décodeur, des unités LSTM et un mécanisme d'attention qui réduit la scalabilité.

2.2 Modèle du langage pour la génération de phrases avec contraintes

Ils prévoient d'utiliser et de construire un plus grand jeux de données que le jeux de données de météo en extrayant toutes les biographies sur Wikipedia [2]. Les approches basées sur les templates ne conviennent plus car il y a 400 000 mots et 700 000 biographies dans le nouvel ensemble de données et qu'ils ne puissent pas définir en détail toutes les règles! Ainsi, ils utilisent le **modèle de langage conditionné par tableau** pour contraindre la génération de la première phrase de l'introduction des articles Wikipédia.

2.2.1 Action de copie

L'action de copie (Copy action) est inspirée du fait que les phrases qui expriment des faits d'un tableau donné copient souvent des mots du tableau. Ils peuvent donc utiliser d'informations de tableau telles que: Q (Tous les tokens de tableau) lors du calcul du score pour des mots de sortie ω_t.

2.2.2 Conditionnement local

Pour utiliser les informations d'occurrence du mot dans le tableau, ils utilisent un descripteur de l'occurrence de mot sous forme de triplet qui inclut la position du mot compté à la fois depuis le début (+) et la fin du champ (-): $z_w = (f_i, p_i^+, p_i^-)_{i=1}^m$.

Ces descripteurs de table $z_{ct} = z_{w_{t(n1)}}, \ldots, z_{w_{t1}}$ sont aussi appelés comme variables de conditionnement locales puisqu'elles décrivent les informations des relations de contexte local (mot précédent) avec la table.

input text (c_t, z_{c_t})

	John	Doe	(	18	April	1352	)	is	a
c_t	13944	unk	17	37	92	25	18	12	4
z_{c_t}	(name,1,2)	(name,2,1) (spouse,2,1) (children,2,1)	∅	(birthd.,1,3)	(birthd.,2,2)	(birthd.,3,1)	∅	∅	∅

figure 2.2.2 - Caractéristiques locales d'un tableau

2.2.3 Conditionnement global

g_f, g_ω représentent des informations de tous les tokens et champs de tableau, tels que les noms d'équipes, les noms de ligue. Ils peuvent aider le modèle à donner une meilleure prédiction.

Ils ont ainsi un nouveau modèle NLM conditionné: P($\omega_t|c_t, z_{ct}, g_f, g_\omega$), avec c_t le contexte à l'instant t et ω_t les candidats de sortie, $\omega \in W \ \ (words \cup CopyAction)$.

Table (g_f, g_w)

name	John Doe
birthdate	18 April 1352
birthplace	Oxford UK
occupation	placeholder
spouse	Jane Doe
children	Johnnie Doe

figure 2.2.3 - Caractéristiques globales d'un tableau

2.3 Approches proposées

Les embeddings de mots pour le contexte et les candidats de sortie $E \in R^{|W|*d}$ est une matrice de paramètres en convertirant l'entrée c_t en $(n-1)$ vecteurs de dimension d. E peut être initialisé de manière aléatoire ou pré-entraîné par un modèle basé sur des embeddings.

Ils apprennent également les embeddings de tableau pour le conditionnement local (descripteurs de table $Z = \{Z^+, Z^-\} \in R^{|F|*l*d}$ et le conditionnement global ($G_{field} \in R^{|F|*g}$, $G_{word} \in R^{|W|*g}$). Notons que G_{word} partage ses informations avec E.

Il est tout à fait possible de voir qu'une instance (un mot) a plusieurs embeddings. Pour la raison que la taille de l'entrée de réseau de neurones doit être fixée, ils agrégent les embeddings par la méthode "**Composant-Wise Max**" pour obtenir les meilleures caractéristiques de toutes les occurrences de ce mot.

2.4 Cadre du modèle

Les entrées du NN sont définis par $x = \{c_t, z_{ct}, g_f, g_\omega\}$ et embeddings de x: $\psi(x) = \{\psi(c_t), \psi(z_{ct}), \psi(g_f), \psi(g_\omega)\}$. Ensuite, une **transformation linéaire** est appliquée sur les entrées obtenir alors les représentations cachées h(x).

Troisièmement, ils utilisent h(x) pour obtenir **2 scores**: le score lié aux mots de sortie possibles (mots du vocabulaire) et le score lié aux descripteurs de tableau. Chaque mot $w \in W \cup Q$ reçoit ainsi un score final en additionnant les 2 scores. Enfin, la fonction **softmax** sera utilisée pour obtenir la probabilité des mots. Ce modèle de langage neuronal est entraîné pour **minimiser la log-vraisemblance négative** d'une phrase d'apprentissage S avec une descente de gradient stochastique.

2.5 Expérimentations

Toutes les biographies sont extraites de Wikipédia en utilisant la dataset WikiBio qui contient plus de 700k biographies, répartis en ensembles de train (80%), de validation (10%) et de test (10%). De chaque biographie, ils ont extrait l'Infobox et la section d'introduction de l'article de wikipedia. Pour les expériences, seule la première phrase de la section d'introduction est générée. L'évaluation se fait en utilisant la méthode **Perplexité** et trois paramètres pour la qualité de la génération: **BLEU-4**, **ROUGE-4** (F-mesure), **NIST4**.

2.6 Résultats

Le modèle de base est le modèle linguistique Kneser-Ney (KN) (5-gram). Le modèle formé est un modèle linguistique de 11-gram. Sans l'utilisation des actions de copie, le modèle fonctionne mal par rapport au modèle KN. En ajoutant une addition locale (Local additioning), la mesure de la perplexité s'améliore légèrement. Les expériences avec des actions de copie et des conditionnements (locaux et globaux) ont les meilleurs scores pour toutes les mesures par rapport aux autres expériences et au modèle de base. L'utilisation de Beam Search pour le décodage des phrases, permet d'explorer un plus grand ensemble de phrases par rapport à la simple recherche gourmande. En comparant différents réglages de Beam, la meilleure validation BLEU peut être obtenue avec une taille de Beam K = 5, et ne prend que 200 ms par phrase.

Parameter	Value		
# word types	$\|\mathcal{W}\|$	=	20,000
# field types	$\|\mathcal{F}\|$	=	1,740
Max. # tokens in a field	l	=	10
word/field embedding size	d	=	64
global embedding size	g	=	128
# hidden units	nhu	=	256

Table 3: Model Hyperparameters

figure 2.6 - Hyperparmètres choisis dans tous les modèles.

3 Limitations

Le modèle linguistique proposé dans cet article peut générer des phrases en introduisant les **actions de copie** à partir du tableau et en conditionnant les champs et les mots du tableau. Le conditionnement local et global améliorent ce modèle dans une large mesure: 15 fois de score BLEU plus qu'un modèle de langage Kneser-Ney. Cependant, ce travail (1) se concentre seulement sur la génération de la première phrase. De plus, (2) la fonction de perte de l'entraînement actuel ne pénalise pas explicitement le modèle pour la génération de faits incorrects. Une fonction de perte qui pourrait évaluer l'exactitude des faits améliorerait certainement la génération des phrases. En outre, (3) l'entrainement du modèle est très lent (plus de 30 heures) et c'est due à la grande quantité de données d'entrainement. Alors, c'est important de choisir les paramètres les plus optimaux en terme de temps de calcul et de précision.

4 Perspectives

Comme indiqué dans la partie de limitations, l'auteur a uniquement concentré sur la génération de la première phrase. Nous réfléchissons ainsi comment obtenir une génération de biographies plus longues? Actuellement, ce modèle génère des textes en agrégeant les conditionnements de tableaux par Composant-Wise Max. Nous pouvons certainement apprendre un encodage (RNN,CNN ou auto-encodeur) pour agréger les conditionnements avec des stratégies plus appropriées. Dans ce cas là, nous aurons des représentations plus riches en tant que les entrées de modèle.

Afin d'améliorer davantage la performance du modèle, nous avons pensé à utiliser d'autres word-embeddings pré-entraînés tels que GloVe[1][3], fastText[2][4] et word2vec[3]. Nous trouvons que ces word embeddings n'ont pas les mêmes structures que ceux de Remi Lebret[4]. De plus, les word embeddings utilisés dans ce modèle sont générés par un outil "***HPCA***", créé par Remi Lebret sous langage C, sur le corpus WikiProject Biography (WikiBio). Par conséquent, si on veut intégrer d'autres word embeddings, il nous faut utiliser ***HPCA*** et une base de données qui ressemble à WikiBio.

Pour pouvoir prouver la généralisation de modèle, nous utilisons souvent plusieurs bases de données et comparons leurs résultats. Comme Remi Lebret a créé cette base de données sur WikiBio en intégrant en plus les informations de tableaux (infobox), cette structure de base de données est vraiment inédite. Nous ne trouvons aucun jeux de données qui a la même structure sur Internet. Dans ce cas là, nous devons creer les scripts qui peuvent traiter les bases de données dans la structure que nous voulons.

References

[1] R. Lebret, D. Grangier, and M. Auli, "Neural Text Generation from Structured Data with Application to the Biography Domain ," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2016.

[2] R. Lebret, D. Grangier, and M. Auli, "WikiBio (Wikipedia Biography Dataset) ," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2016.

[3] C. D. M. Jeffrey Pennington, Richard Socher, "GloVe: Global Vectors for Word Representation ,"

[4] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *arXiv preprint arXiv:1607.04606*, 2016.

[1] Wikipedia 2014 + Gigaword 5 (6B tokens, 400K vocab, uncased, 50d, 100d, 200d, 300d vectors, 822 MB download)

[2] 1 million word vectors trained on Wikipedia 2017, UMBC webbase corpus and statmt.org news dataset (16B tokens).

[3] Pre-trained vectors trained on part of Google News dataset (about 100 billion words).

[4] L'auteur de Neural Text Generation from Structured Data with Application to the Biography Domain