Passé, présent, futurs : induction de carrières professionnelles à partir de CV

Charles-Emmanuel Dias* — **Vincent Guigue*** — **Patrick Gallinari***

* *Sorbonne Université, UPMC Univ Paris 6, UMR 7606, LIP6, F-75005, Paris, France – Mail: Prénom.Nom@lip6.fr*

RÉSUMÉ. *L'extraction, la structuration et l'exploitation d'informations à partir de données textuelles brutes est une tâche complexe. L'apprentissage de représentation permet de dépasser la barrière syntaxique que représente le codage textuel et d'encoder les informations selon des règles définies. Les dernières avancées en apprentissage statistique ont permis d'améliorer considérablement l'analyse sémantique des textes. Dorénavant, le principal verrou technologique se déplace vers le raisonnement afin d'extrapoler des connaissances non-explicites. Dans cet article, nous nous intéressons à l'apprentissage d'un espace latent sur un large corpus de CV pour l'induction de carrières professionnelles. Nous proposons d'abord un modèle de normalisation sémantique basé sur une architecture encodeur/décodeur pour réduire la diversité de dénomination des métiers puis nous développons des opérateurs de raisonnement pour modéliser la hiérarchie dans les organigrammes d'entreprises et prédire les enchaînements de postes dans les CV.*

ABSTRACT. *Extracting, structuring and exploiting information from freeform text is a difficult task. Learning embeddings with chosen properties and going beyond simple syntax encoding contributed to significant improvements in semantic analysis. Recently, the focus has shifted from word and document embeddings to reasoning in order to infer or predict new knowledge. In this paper, we focus on job & educational background embeddings that are learned from a large CV corpus. We aim at modeling users careers and forecasting their choices. Inspired by recent work in machine translation, we design a Recurrent Neural Network architecture to normalize job & qualification titles. Once this semantic step achieved, we build another RNN to predict position chaining in CV.*

MOTS-CLÉS : *Apprentissage de représentations, Réseaux de neurones, Recommandation.*

KEYWORDS: *Embeddings, Neural networks, Recommandation.*

1. Introduction

Extraire, structurer et exploiter efficacement de l'information à partir de textes libres est actuellement un défi majeur. L'apprentissage de représentation permet de dépasser la barrière syntaxique que représente le texte et d'encoder les informations dans un espace vectoriel. Cette approche a permis des avancées significatives dans plusieurs applications autour du traitement des données textuelles. Sur le plan sémantique, les algorithmes *Word2Vec* ou *GloVe* ont introduit une approche de modélisation locale très pertinente (Mikolov *et al.*, 2013 ; Pennington *et al.*, 2014). Les techniques de traduction automatique sont en train de changer de paradigme ; les approches d'alignement structuré type HMM ou CRF (Lafferty *et al.*, 2001) sont dépassées par les réseaux de neurones récurrents (RNN) où un groupe de mots est factorisé dans une représentation latente avant d'être re-généré dans une nouvelle langue selon une logique encodeur/décodeur (Cho *et al.*, 2014). Les techniques d'apprentissage de représentation ont également démontré leur potentiel en gestion des connaissances: l'encodage des concepts et la définition d'opérateurs dans les espaces latents permettent d'exploiter de larges bases et d'inférer de nouvelles connaissances tout en étant robuste au bruit (Bordes *et al.*, 2011). Les systèmes les plus récents intègrent l'ensemble de la chaîne: l'analyse du texte brut, la représentation des connaissances et l'inférence de nouvelles relations (Weston *et al.*, 2013).

Dans cet article, nous nous intéressons à l'apprentissage d'un espace latent textuel pour la représentation des parcours professionnels extraits de *curriculum vitae*. A partir d'un large corpus de CV, nous souhaitons comprendre les évolutions de carrières et proposer un outil de recommandation pertinent. Une carrière n'est pas instantanée mais une séquence d'événements (Savickas, 2005). Les parcours professionnels suivent aussi une structure hiérarchique particulière : statistiquement, les personnes avancent progressivement vers les diplômes et les postes les plus élevés. Nos représentations doivent capter ces propriétés hiérarchiques afin d'exploiter efficacement l'historique des parcours et induire leur suite. L'enjeu est donc double: apprendre des représentations pertinentes des formations et des métiers sur une population, à la manière du filtrage collaboratif ; puis proposer une modélisation individualisée des évolutions de carrière pour dépasser les simples statistiques d'enchaînements de postes présents dans la base de données. Le premier verrou est d'ordre sémantique. Bien que structuré, un CV est avant tout constitué de texte libre ; de ce fait, un même élément peut être représenté par une multitude d'intitulés, syntaxes et abréviations, contenant régulièrement des fautes. Les services de gestion des ressources humaines utilisent habituellement des référentiels métier/compétence pour normaliser les CV. Ces référentiels sont statiques et présentent des lacunes ; nous proposons de construire un modèle de langue pour s'abstraire de la barrière syntaxique et obtenir des représentations conceptuelles. Orthogonalement, nous proposons de travailler sur les aspects hiérarchiques et le modèle prédictif personnalisé, dans l'espace latent appris. Avec un horizon temporel de 1, le modèle donnera simplement l'évolution de carrière la plus logique, correspondant aux échelons hiérarchiques ; En élargissant l'horizon temporel,

nous prendrons en compte l'historique d'une personne pour affiner sa probable future position.

Nous proposons d'adapter les méthodes d'encodeur-décodeur issues de la traduction automatique (Cho *et al.*, 2014) pour construire un espace latent textuel intégrant de bonnes propriétés sémantiques et syntaxiques. Ces techniques à base de réseaux de neurones récurrents permettent notamment d'encoder caractère par caractère un champ textuel et décoder la représentation obtenue en une autre chaîne de caractères. Le décodeur donne une interprétation claire de l'espace de représentation, en langage naturel. Il s'agit de l'étape d'extraction d'information de notre système. La partie raisonnement se résume à un opérateur de modélisation hiérarchique qui est implémenté par un second réseau de neurones récurrents. Cet opérateur prend un ou plusieurs éléments de l'historique selon la tâche visée.

Cet article est organisé de la façon suivante: tout d'abord, nous exposons le contexte bibliographique de nos travaux (section 2). Ensuite, nous donnons les détails de l'architecture retenue pour l'extraction d'information et l'analyse de carrière (section 3.1). Enfin, nous évaluons notre système sur différentes tâches pour caractériser ses capacités de modélisation de la langue et de prédiction du comportement des utilisateurs (section 4).

2. État de l'art

Dans cette section nous présentons différents travaux relatifs à l'apprentissage de représentations textuelles, aux réseaux de neurones récurrents et à l'architecture d'encodage/décodage ainsi que leurs liens avec notre contexte applicatif.

2.1. *Apprentissage de Représentations Textuelles*

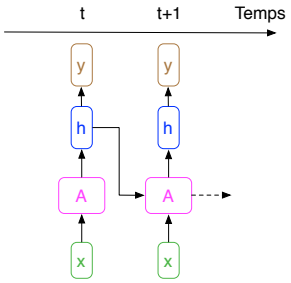
En traitement automatique du langage naturel, les avancées récentes sont étroitement liées aux modèles de langues probabilistes (Bengio *et al.*, 2003). Les modèles contextuels, déterminant la représentation d'un mot par rapport à son contexte local, ont permis de franchir un palier qualitatif ces dernières années ; les algorithmes *Word2Vec* (Mikolov *et al.*, 2013) et *GloVe* (Pennington *et al.*, 2014) sont les implémentations les plus populaires de ce paradigme. Ces techniques encodent remarquablement la sémantique, les représentations des mots synonymes étant très proches, mais leurs propriétés vont plus loin. Plusieurs opérateurs linéaires permettent d'inférer les représentations des féminins (e.g. $vec(\text{"}king\text{"}) - vec(\text{"}man\text{"}) + vec(\text{"}woman\text{"}) \approx vec(\text{"}queen\text{"})$) ou des pluriels. Le raisonnement, c'est-à-dire la capacité à inférer de nouvelles représentations, devient un élément clé dès les premières étapes de l'analyse textuelle: c'est le début de la fusion entre l'analyse du texte brut, l'extraction d'informations et l'inférence de connaissances.

Néanmoins, ces modèles de langues probabilistes travaillent au niveau des mots. Ils sont donc limités par leur dictionnaire. Pour lever cette contrainte et gérer tous les

mots (ainsi que les fautes d'orthographe), il est possible d'apprendre des modèles de langues caractère par caractère (Sutskever *et al.*, 2011). En effet, les réseaux de neurones récurrents (RNN) sont capables de capter efficacement les dépendances longues et permettent de modéliser de manière cohérente des séquences (Graves, 2013). Ils peuvent même générer un texte parfaitement intelligible (Karpathy *et al.*, 2015). Afin de différentier les concepts très précis des mots polysémiques, (Vilnis et McCallum, 2014) utilisent des distributions gaussiennes à la place de vecteurs continus. Cette idée fonctionne également au niveau de l'encodage de classes multiples pouvant se superposer partiellement ou s'imbriquer (Dos Santos *et al.*, 2016).

2.2. *Réseau de neurones récurrents*

Les réseaux de neurones récurrents (RNN) présentent l'avantage de traiter naturellement des séquences de longueurs variables, ce qui est bien adapté à la modélisation des données textuelles. En effet, contrairement aux réseaux de neurones classiques, le calcul d'une sortie y_t à un l'instant t prend en compte à la fois l'entrée actuelle x_t et une représentation des entrées précédentes h_{t-1} (Elman, 1990)



$$\begin{cases} h_t = \sigma_h(W_h x_t + U_h h_{t-1} + b_h) \\ y_t = \sigma_y(W_y h_t + b_y) \end{cases} \quad [1]$$

C'est cet état caché h_t qui est une composition non linéaire de l'ensemble des entrées $x_1, .., x_t$ modulée par une fonction d'activation σ_h qui permet de conserver une mémoire de la séquence, il s'agit donc d'une représentation latente de l'ensemble du passé. Néanmoins, cette version classique du réseau de neurones récurrent souffre du problème de disparition du gradient ; en effet, celui-ci décroit de manière exponentielle du fait des nombreuses multiplications lors de la rétropropagation temporelle, ce qui rend difficile l'apprentissage sur de longues séquences. Face à ce problème, (Le *et al.*, 2015) montrent qu'une initialisation spécifique, combinée à des fonctions d'activations *ReLU*[1] rend l'apprentissage de dépendances plus longues possibles. Cependant, les meilleurs résultats sont obtenus en modifiant l'architecture afin de diminuer l'enchaînement des multiplications (Mikolov *et al.*, 2014). C'est dans ce contexte qu'ont été développées les cellules "à portes".

Les deux cellules à portes les plus connues sont le LSTM (Long Short Term Memory) (Hochreiter et Schmidhuber, 1997) et sa variante simplifiée, la GRU (Ga-

1. $f(x) = \max(0, x)$

ted Recurrent Unit) (Cho *et al.*, 2014). (Colah, 2015) expose en détail l'intuition et le fonctionnement de ces cellules. En règle générale et en particulier sur les données textuelles, les cellules GRU et LSTM offrent des performances similaires (Greff *et al.*, 2015 ; Zaremba, 2015 ; Chung *et al.*, 2015). Nous privilégions donc la GRU pour nos travaux car elle est plus simple.

Dans ce type d'architecture, nous introduisons la notion d'état s auquel est ajouté ou retiré de l'information, cet état s_t correspond à la sortie de la cellule à l'instant t ; il est donc équivalent au y_t de la cellule RNN ci-dessus. Dans une cellule GRU à l'instant t, l'état de sortie s_t est une combinaison additive (via une porte $g_t \in [0, 1]$) de l'état précédent s_{t-1} et d'un statut actuel $\hat{s_t}$ (où $\odot$ désigne la multiplication terme à terme). La cellule GRU doit être vue comme une sorte de mémoire qui s'incrémente sélectivement. La porte de "mise à jour" g_t permet de choisir comment combiner le passé (s_{t-1}) avec l'apport d'information présent $\hat{s_t}$.

$$s_t = g_t \odot s_{t-1} + (1 - g_t) \odot \hat{s_t} \quad [2]$$

Le statut actuel $\hat{s_t}$ est une combinaison de l'entrée x_t et d'une partie de l'état précédent s_{t-1} (filtré par une porte $r_t \in [0, 1]$).

$$\hat{s_t} = \phi(W^s[(r_t \odot s_{t-1}); x_t] + b^s) \quad [3]$$

Ce statut actuel $\hat{s_t}$ est donc une interprétation de l'entrée x_t tenant compte du passé s_{t-1}. La porte r_t est dite porte de "redémarrage", puisqu'elle est capable d'effacer le passé et choisit quoi ajouter à l'état s_t. Les deux portes g_t et r_t sont apprises puis appliquées sur l'entrée x_t et l'état précédent s_{t-1}:

$$\hat{g_t} = \sigma(W^g[s_{t-1}; x_t] + b^g) \quad [4]$$

$$\hat{r_t} = \sigma(W^r[s_{t-1}; x_t] + b^r) \quad [5]$$

Par ailleurs, si nous saturons r_t à 1 et g_t à 0 alors la GRU est équivalente au RNN simple présenté auparavant.

2.3. *Architecture encodeur/décodeur*

En traduction automatique, la tâche principale consiste à passer d'une phrase dans la langue a, p_a à sa contrepartie dans la langue b, p_b. Les approches à base d'alignement mot à mot ont cédé la place ces dernières années à des approches plus globales basées sur une architecture encodeur/décodeur. Soit deux RNN: $(encoder, decoder)$, l'encodeur prend en entrée la phrase p_a et crée une représentation z_p de celle-ci. Cette représentation est ensuite décodée en une phrase $\hat{p_b}$ du langage cible. La représentation intermédiaire doit capter le sens de la phrase, la qualité du décodeur garantit l'intelligibilité de la proposition dans la langue b.

$$z_p = encoder(p_a), \qquad \hat{p_b} = decoder(z_p) \quad [6]$$

Différentes variantes de cette architecture existent. Il est possible de décoder la phrase cible p_b directement (Sutskever *et al.*, 2014), mot à mot (Cho *et al.*, 2014) ou caractères par caractères (Lee *et al.*, 2016). Un mécanisme d'attention (Bahdanau *et al.*, 2014) peut également être ajouté afin d'aligner automatiquement les mots de la phrase source avec la cible.

3. Espace latent de carrières professionnelles

Nous décrivons dans un premier temps notre modèle d'apprentissage de représentation basé sur une architecture d'encodeur/décodeur ainsi que les détails liés à l'apprentissage du modèle. Nous décrivons ensuite le modèle hiérarchique de prédiction du prochain poste.

3.1. *Encodeur - Décodeur: Modèle de langue syntaxique et sémantique*

Nous proposons d'adapter les méthodes d'encodeur-décodeur issues de la traduction automatique (Cho *et al.*, 2014) pour construire un espace latent de carrières professionnelles. La force des techniques à base de réseaux de neurones récurrents est de pouvoir encoder directement caractère par caractère un champ textuel. Cette faculté est particulièrement importante ici puisque nous devons unifier des représentations syntaxiques faites de différents termes, abréviations voire fautes d'orthographe. Dans un second temps, le décodeur offre la possibilité de réinterpréter un concept de l'espace latent en générant une chaîne de caractères ; il est aussi basé sur un réseau de neurones récurrents.

Encodeur. Nous avons opté pour une architecture issue de la traduction automatique car la problématique est la même : dans un texte libre $\mathbf{txt}$, non-normalisé, plusieurs graphies $\{\mathbf{txt}_1, \mathbf{txt}_2, ...\}$ peuvent correspondre à la même signification. Le but est donc d'apprendre une représentation unifiée $\mathbf{z} \in \mathbb{R}^d$ captant le sens général de la phrase, ainsi qu'une fonction pour passer d'un texte à sa représentation, avec la propriété suivante: $\forall n,\ f(\mathbf{txt}_n) \approx \mathbf{z}$. Cette fonction correspond à l'encodeur. Comme le montre la figure 1, l'unité de base du système est la lettre, passée à l'encodeur dans un format *1-hot*: un vecteur de la taille de l'alphabet $\mathcal{A}$ rempli de 0, sauf à la position du caractère visé. Deux caractères spéciaux ^ et \$ encodent respectivement le début et la fin de séquence. Un texte $\mathbf{txt}$ correspond donc à un ensemble de lettres $\{\mathbf{x}_1, \mathbf{x}_2, \ldots\}$ avec $\mathbf{x}_{lettre} \in \{0,1\}^{|\mathcal{A}|}$. Comme cela a été montré en section 2, notre architecture permet d'encoder l'information lue dans un vecteur de $\mathbf{z} \in \mathbb{R}^d$ ($d = 256$ dans le cadre de cet article).

Dans le détail, l'encodeur est constitué de deux cellules GRU: l'une encodant le texte de gauche à droite $\overrightarrow{G}_r$ et l'autre dans le sens inverse $\overleftarrow{G}_l$. Les vecteurs $\mathbf{h}$ issus de cet encodage sont par la suite concaténés et combinés via une transformation li-

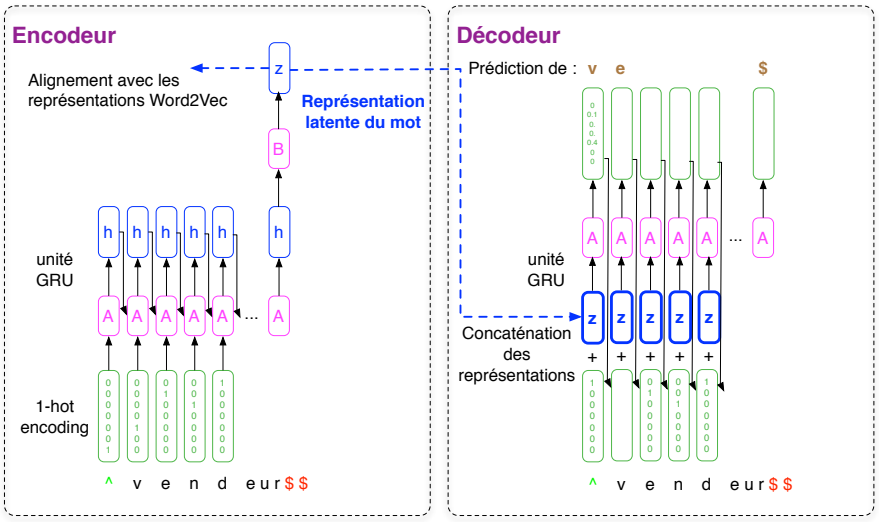


Figure 1. *Modèle encodeur-décodeur. L'encodeur est un Bi-GRU dont les sorties sont concaténées et transformées linéairement. Le décodeur est une cellule GRU simple qui prédit le caractère suivant à partir du caractère précédent concaténé au vecteur encodé. La représentation doit également pouvoir prédire un contexte issu de représentations Word2Vec.*

néaire pour obtenir $\mathbf{z}$. Cette technique de Bi-RNN (Schuster et Paliwal, 1997) permet d'apprendre de meilleures représentations.

$$encode(\mathbf{txt}) = \mathbf{z} = \sigma(W^e[\overleftarrow{G_l(\mathbf{txt})}; \overrightarrow{G_r(\mathbf{txt})}]) \quad [7]$$

Nous disposons systématiquement de l'ensemble de l'intitulé à encoder: cette approche bidirectionnelle ne remet donc pas en cause les capacités du système en inférence.

Décodeur. Le décodeur est également un réseau de neurones récurrents, dans lequel nous jouons sur l'entrée pour garantir la signification du résultat. A chaque pas de temps, nous fournissons le code $\mathbf{z}$ du concept, concaténé avec la représentation d'une lettre (au format *1-hot*). Nous sommes ainsi en mesure de prédire la prochaine lettre la plus vraisemblable étant donné le contexte $\mathbf{z}$.

Pour générer du texte, nous partons du caractère *neutre* ^ (codant le début de séquence) ; le début de la séquence dépend donc principalement du contexte ; ensuite, le modèle de langue appris garantit l'intelligibilité du texte généré. Nous nous arrêtons lors de la prédiction du caractère $.

Apprentissage. Dans le détail, les CV sont anonymisés et pré-traités ; chaque ligne (professionnelle ou cursus scolaire) est divisée en un intitulé et une description (en-

semble de mots de contexte). Pour l'apprentissage, chaque intitulé est transformé en une suite de vecteurs 1-hot $\{\mathbf{x}_1, \mathbf{x}_2, \ldots\}, \mathbf{x}_i \in \{0,1\}^{|\mathcal{A}|}$ représentant l'enchaînement des caractères. Les séquences sont limitées à 32 caractères pour des raisons computationnelles, environ 5% des intitulés sont donc tronqués. L'encodeur et le décodeur sont appris simultanément par descente de gradient stochastique temporelle modifiée selon la technique de RMSProp (Hinton *et al.*, 2012).

Afin que ces représentations $\mathbf{z}$ soient pertinentes, nous combinons différents critères d'apprentissage. Nous utilisons d'abord des représentations externes pour stabiliser et accélérer l'apprentissage: nous rapprochons les $\mathbf{z}$ d'un modèle pré-entrainé sur un large corpus avec l'algorithme Word2Vec[2]. Nous cherchons ensuite à reconstruire les intitulés de postes à l'identique, à la manière d'un auto-encodeur: les corrections ont alors lieu à la fois sur l'encodeur et le décodeur. Enfin, nous cherchons à reconstruire le contexte de l'intitulé de poste à partir de l'intitulé lui-même afin de rapprocher les représentations des synonymes et donner une véritable dimension sémantique à la représentation. La philosophie est la même que pour le *negative sampling* de l'approche Word2Vec.

3.2. *Prédiction de parcours professionnels*

La prédiction du prochain item (travail ou cursus scolaire) dans un CV est une tâche très difficile pour deux raisons principales. La barrière syntaxique, à travers la personnalisation des intitulés, empêche de capitaliser l'expérience entre les utilisateurs. La nature même des carrières pose ensuite problème, la majorité des transitions renvoyant vers un poste équivalent ou un changement de contexte mais plus rarement vers une promotion. Nous avons abordé la variabilité de la syntaxe dans la section précédente, nous nous intéressons ici à la modélisation des hiérarchies de postes et à la prédiction des évolutions de carrières des utilisateurs.

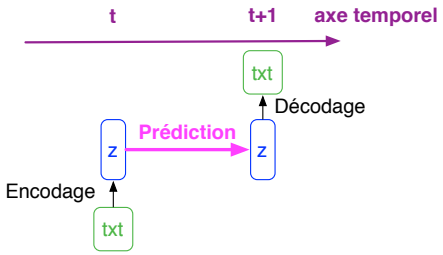


Figure 2. *Architecture du prédicteur de prochain intitulé (travail ou cursus scolaire).*

Le prédicteur de prochain item est de nouveau un réseau de neurones récurrents avec une couche GRU. Pour étudier la topologie de l'espace des items, nous pouvons

2. Corpus `frWac` disponible: `http://fauconnier.github.io/#data`

# Intitulés conservés:	5000	3000	2000
# Intitulés de poste inconnus	443 610 (15%)	536 520 (18%)	631 120 (21%)
# Diplômes inconnus	737 955 (55%)	782 484 (58%)	817 517 (61%)

Tableau 1. *Nombre et proportion d'intitulés inconnus dans l'ensemble des parcours en fonction du nombre d'intitulés uniques fréquents conservés.*

lui fournir une seule entrée $\mathbf{z}_t$; nous obtenons alors le comportement $\mathbf{z}_{t+1}$ dominant à partir de ce point de départ. Le même réseau est capable d'assimiler une séquence d'items passés $S_u = [\mathbf{z}_1, \ldots, \mathbf{z}_t]$ pour prédire $\mathbf{z}_{t+1}$. La sortie est alors personnalisée par rapport à l'ensemble du parcours d'un utilisateur u. Ce système conserve et apprend l'ordonnancement des items mais pas les dates ; le fait de rester longtemps sur un poste n'est pas encore pris en compte.

Nous obtenons donc un système global composé de deux RNN disposés orthogonalement: l'encodeur/décodeur et le modèle de prédiction du prochain item. L'ensemble des CV est utilisé pour apprendre le premier RNN, c'est-à-dire le modèle de langue. Le prédicteur est lui classiquement évalué en validation croisée (5 *folds*): une partie des CV servant à apprendre le modèle et l'autre à l'évaluer.

4. Expériences

Dans cette section, après une présentation des données utilisées, nous évaluons les propriétés et l'utilité de nos représentations. Nous montrons dans un premier temps les capacités syntaxiques et sémantiques de notre système d'encodage décodage. Puis, nous montrons son intérêt en cherchant à prédire la suite des parcours professionnels.

4.1. *Données extraites des curriculum vitae*

Nous avons à notre disposition un ensemble de 915 925 *curriculum vitae* anonymisés et pré-traités au format XML. Chaque CV devient ainsi une paire de séquences: le parcours éducatif et le parcours professionnel. Les parcours vides, incomplets ou de longueur supérieurs à 12 sont écartés car ils correspondent généralement à un problème de traitement des CV bruts avec un résultat inintelligible. Après ce filtrage, il reste 656 134 paires de parcours éducatifs et professionnels. Le parcours éducatif est composé d'une séquence de noms de diplômes, triée par date d'obtention (du plus vieux au plus récent). Le parcours professionnel est quant à lui composé d'une séquence de titres de postes associés à leurs descriptions, triée par date de début de poste (du plus vieux au plus récent). Les noms de diplômes, les intitulés de postes et leurs descriptions associées sont extraits directement du texte des CV (sans traitement linguistique).

Après avoir enlevé les espaces redondants, les accents et les majuscules, il reste 266 622 intitulés de postes et 552 439 diplômes distincts. A titre de comparaison, le référentiel métier le plus complet actuellement (ESCO) compte 4800 entrées. En apprentissage statistique, l'approche la plus classique consiste à conserver uniquement les N intitulés/noms les plus fréquents et de remplacer les autres par un même identifiant *inconnu*. Malheureusement, le tableau 1 montre que la distribution des intitulés de postes et surtout de diplômes est trop uniforme: la proportion d'*inconnus* est trop élevée avec ce système. Les variations de formulation, les abréviations et les flexions sont trop nombreuses: c'est ce qui nous a poussé à développer le modèle de normalisation linguistique.

4.2. *Propriétés syntaxiques et sémantiques*

Nous souhaitons apprendre un encodeur robuste. De ce fait, l'encodeur doit être insensible aux différentes graphies possibles d'un même mot ; en d'autres termes, un mot et ses variantes mal orthographiées doivent avoir des représentations proches. Pour évaluer cette propriété nous procédons à une expérience simple : nous suivons l'évolution d'une représentation (par rapport à son original) en fonction de modifications orthographiques additives, soustractives et commutatives. Les représentations étant en grande dimension (256), nous utilisons la similarité cosinus normalisée :

$$\text{sim}(\mathbf{z}, \mathbf{z}_{mod}) = \frac{1}{2}\left(\frac{\mathbf{z}\cdot\mathbf{z}_{mod}}{\|\mathbf{z}\|\|\mathbf{z}_{mod}\|} + 1\right) \in [0,1] \qquad [8]$$

La figure 3 (gauche) montre la robustesse de l'encodeur: jusqu'à environ 4 modifications, la similarité reste supérieure à 80%. Ce résultat est important: une seconde illustration (figure 4) montre que des représentations proches mènent à la même reconstruction.

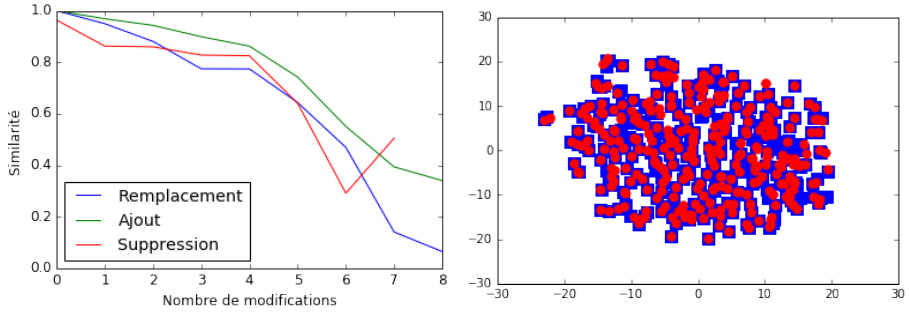


Figure 3. *(à gauche) Évolution de la similarité cosinus normalisée (eq. 8) du vecteur bruité par rapport à l'original en fonction de différentes modifications syntaxiques. (à droite) Projection T-SNE des représentations de métiers masculins (carrés bleus) et de leurs équivalents féminins (ronds rouges)*

```
^manutention$                 => ^manutentionnaire$
^manutencion$                 => ^manutentionnaire$
^manutentionaire$             => ^manutentionnaire$
^technicien$, ^technicienne$  => ^technicien$, ^technicien$
^vendeur$, ^vendeuse$         => ^vendeuse$, ^vendeuse$
^assistant$, ^assistante$     => ^assistant$, ^assistant$
```

Figure 4. *Processus d'encodage/décodage de différentes flexions d'un intitulé: la reconstruction est identique.*

Nous nous sommes également intéressés au problème du genre: en effet, la plupart des métiers ont une version féminine de leur intitulé. Celle-ci se limite souvent à des modifications mineures d'orthographe, mais pas toujours. Comme pour l'expérience précédente, l'enjeu consiste à avoir des représentations proches. Pour vérifier expérimentalement cela, nous disposons d'une liste de 312 métiers féminisés[3]. En projetant les représentations des métiers masculins (carrés bleus) et féminins (rond rouges) dans un espace bi-dimensionnel grâce à l'algorithme T-SNE (Maaten et Hinton, 2008), nous montrons à quel point les deux déclinaisons d'un même métier ont des représentations proches.

```
|Mot encodé:                        |
|^responsable clientele internat$   |
|-----------------------------------  |
| ('^responsable de clientele$'),   |
| ('^responsable commerciale$'),    |
| ('^responsable clientele$'),      |
| ('^responsable cliente$'),        |
| ('^responsable$')                 |
```

Figure 5. *Exemple de décodage de la représentation de "responsable clientèle internationale". Les sorties affichées sont les cinq plus probables.*

Afin d'illustrer l'intérêt du décodeur, nous proposons en figure 5 un exemple de processus standard: un intitulé est projeté dans l'espace latent puis décodé. Ces décodages correspondent à des interprétations possibles de la représentation normalisée dans l'espace latent. Le décodeur fonctionne ici en mode stochastique, ce qui explique que la réponse ne soit pas unique: les résultats présentés sont les plus redondants. Le décodeur ne propose pas de fonction de score fiable pour classer ces résultats: comme avec les approches de type markovienne, la longueur des séquences produites fausse toute comparaison.

3. Liste disponible sur: `http://ameliorsonfrancais.com/grammaire/genres/feminisation-des-metiers-et-des-titres/`

4.3. *Evaluation de la prédiction de carrière*

Comme dans la section précédente, nous évaluons les performances directement dans l'espace latent en mesurant l'écart entre la position prédite de l'utilisateur à l'instant $t+1$ et la représentation encodée de l'item de son CV à $t+1$ en utilisant la métrique de l'équation 8. Une telle mesure est certes relativement abstraite, cependant, il est impossible de faire autrement à cause de la variabilité dans les intitulés utilisés par les utilisateurs. Après avoir appris la structure encodeur/décodeur sur l'intégralité de nos données, nous divisons les 656 134 parcours en cinq parts égales ($\approx$ 131 226 parcours chacune) pour faire de la validation croisée. Nous apprenons un RNN sur 4/5ème des données et évaluons son comportement sur la base de test restante. Nos représentations étant partiellement alignées sur des représentations Word2Vec et apprisent avec une philosophie proche, nous comparons les comportements de deux modèles prédictifs dans les deux univers. Plus important: nous avons fait varier la taille de l'historique considéré dans chaque CV. Avec un historique de taille unitaire, nous avons simplement un modèle général d'enchaînement des intitulés ; mais lorsque l'historique augmente, nous obtenons un prédicteur personnalisé compilant les informations sur l'utilisateur pour affiner la proposition.

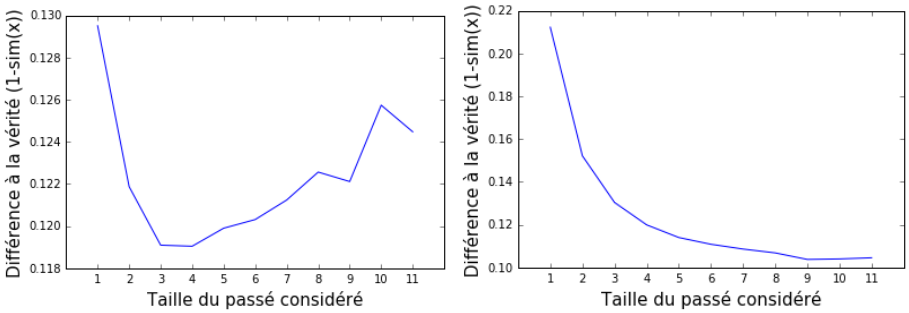


Figure 6. *Courbe de l'erreur de prédiction du prochain emploi d'un opérateur hiérarchique en fonction de la taille du passé considéré (sur le split 1), pour des représentations issues de Word2Vec (gauche) et de notre encodeur (droite).*

	Split 1	Split 2	Split 3	Split 4	Split 5
Encodage W2V	0.529	0.527	0.527	0.527	0.528
Encodage E/D	0.828	0.825	0.825	0.829	0.827

Tableau 2. *Similarités cosinus moyennes entre les prochains emplois prédits et leurs représentations réelles, normalisées par la similarité moyenne entre représentations (W2V $\approx$ 0.25 et E/D $\approx$ 0.08)*

Les résultats sur le split 1 sont présentés en figure 6, la notion de similarité est simplement renversée pour mesurer une erreur: $\text{err}(\mathbf{z}, \mathbf{z}') = 1 - \text{sim}(\mathbf{z}, \mathbf{z}')$. La pre-

```
 t   | vérité:                | prédiction:
-------------------------------------------------------
t(0) | ^assistant$            |
t(1) | ^assistant commercial$ | ^assistante commercial$
t(2) | ^conseiller clientele$ | ^conseiller commercial$
t(3) | ^conseiller clientele$ | ^conseiller commercial$
```

Figure 7. *Exemple de prédiction d'un parcours professionnels*

mière conclusion concerne la capacité d'exploitation de l'historique: les représentations E/D (basées sur l'architecture encodeur/décodeur) permettent de tirer parti de l'ensemble de l'historique pour améliorer la prédiction alors que celles W2V (issues de Word2Vec) plafonnent à un horizon 3. L'axe des ordonnées n'est pas directement interprétable, les représentations évoluant dans 2 espaces distincts non normalisés. Afin de dépasser ce problème de normalisation, nous avons calculé la distance moyenne entre deux intitulés aléatoires et nous avons normalisé les résultats par rapport à cette valeur. Le tableau 2 recense les similarités normalisées entre la prédiction et l'intitulé du prochain item: notre modèle dépasse nettement Word2Vec et les résultats sont stables sur les différentes divisions de la validation croisée. En combinant le modèle unitaire et le décodeur, il est possible d'analyser un déroulé de carrière *type* à partir d'un point de départ donné (soit au niveau des études, soit dans un parcours professionnel). La figure 7 présente un tel résultat, en partant d'un exemple issu de la base de CV et en confrontant les prédictions avec l'évolution effective de l'utilisateur.

5. Conclusion et perspectives

Les parcours professionnels présentent un double enjeu de modélisation : la compréhension de formulations très variées pour décrire des postes similaires et la prise en compte de l'ordonnancement propre à chaque séquence ainsi que la hiérarchie implicite de l'ensemble des postes et diplômes. Nous proposons une double architecture pour faire face à ces verrous: un premier réseau de normalisation sémantique des intitulés utilisés dans les CV, qui agit comme un traducteur, puis un second réseau prédictif captant les évolutions de carrière à la fois au niveau général et au niveau personnalisé.

Nous avons démontré l'efficacité de l'architecture proposée pour l'analyse d'un corpus de CV. Ce travail servira de base pour concevoir des systèmes de recommandation de nouvelle génération pour les personnes à la recherche d'un emploi ; en effet, de nombreuses perspectives applicatives s'offrent à nous, depuis l'aide au remplissage de formulaire, la proposition de positionnement dans les référentiels existants et jusqu'à la sélection de postes correspondant au CV des utilisateurs. Nous serons bientôt en mesure de distinguer les parcours classiques des parcours atypiques et de prendre en compte les durées associées à chaque ligne du CV pour améliorer la modélisation

de l'utilisateur et affiner les propositions qui lui sont faites. Cette étude n'exploite pas encore les représentations des cursus scolaires: un travail reste à fournir pour être capable de faire de la recommandation de premier emploi (démarrage à froid) et pour comprendre l'impact des études sur l'évolution des carrières des utilisateurs.

6. Remerciements

Ce travail a été réalisé en partie avec le soutien du Projet Investissement d'Avenir (PIA) ITER-RH.

7. Bibliographie

Bahdanau D., Cho K., Bengio Y., « Neural machine translation by jointly learning to align and translate », *arXiv preprint arXiv:1409.0473*, 2014.

Bengio Y., Ducharme R., Vincent P., Jauvin C., « A neural probabilistic language model », *journal of machine learning research*, vol. 3, p. 1137-1155, 2003.

Bordes A., Weston J., Collobert R., Bengio Y., « Learning Structured Embeddings of Knowledge Bases. », *AAAI'11*, 2011.

Cho K., van Merrienboer B., Gulcehre C., Bougares F., Schwenk H., Bengio Y., « Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation », *Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, 2014.

Chung J., Gülçehre C., Cho K., Bengio Y., « Gated feedback recurrent neural networks », *CoRR, abs/1502.02367*, 2015.

Colah C., « Understanding LSTM Networks », , `http://colah.github.io/posts/2015-08-Understanding-LSTMs/`, 2015.

Dos Santos L., Piwowarski B., Gallinari P., « Multilabel Classification on Heterogeneous Graphs with Gaussian Embeddings », *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, Springer, p. 606-622, 2016.

Elman J. L., « Finding structure in time », *Cognitive science*, vol. 14, n° 2, p. 179-211, 1990.

Graves A., « Generating sequences with recurrent neural networks », *arXiv preprint arXiv:1308.0850*, 2013.

Greff K., Srivastava R. K., Koutník J., Steunebrink B. R., Schmidhuber J., « LSTM: A search space odyssey », *arXiv preprint arXiv:1503.04069*, 2015.

Hinton G., Srivastava N., Swersky K., « Lecture 6a Overview of mini–batch gradient descent », *Coursera Lecture slides https://class. coursera. org/neuralnets-2012-001/lecture,[Online*, 2012.

Hochreiter S., Schmidhuber J., « Long short-term memory », *Neural computation*, vol. 9, n° 8, p. 1735-1780, 1997.

Karpathy A., Johnson J., Fei-Fei L., « Visualizing and understanding recurrent networks », *arXiv preprint arXiv:1506.02078*, 2015.

Lafferty J., McCallum A., Pereira F., « Conditional random fields: Probabilistic models for segmenting and labeling sequence data », *Proceedings of the eighteenth international conference on machine learning, ICML*, vol. 1, p. 282-289, 2001.

Le Q. V., Jaitly N., Hinton G. E., « A simple way to initialize recurrent networks of rectified linear units », *arXiv preprint arXiv:1504.00941*, 2015.

Lee J., Cho K., Hofmann T., « Fully Character-Level Neural Machine Translation without Explicit Segmentation », *arXiv preprint arXiv:1610.03017*, 2016.

Maaten L. v. d., Hinton G., « Visualizing data using t-SNE », *Journal of Machine Learning Research*, vol. 9, p. 2579-2605, 2008.

Mikolov T., Joulin A., Chopra S., Mathieu M., Ranzato M., « Learning longer memory in recurrent neural networks », *arXiv preprint arXiv:1412.7753*, 2014.

Mikolov T., Sutskever I., Chen K., Corrado G. S., Dean J., « Distributed representations of words and phrases and their compositionality », *Advances in neural information processing systems*, p. 3111-3119, 2013.

Pennington J., Socher R., Manning C. D., « Glove: Global Vectors for Word Representation. », *EMNLP*, vol. 14, p. 1532-43, 2014.

Savickas M. L., « The theory and practice of career construction », *Career development and counseling: Putting theory and research to work*, vol. 1, p. 42-70, 2005.

Schuster M., Paliwal K. K., « Bidirectional recurrent neural networks », *IEEE Transactions on Signal Processing*, vol. 45, n° 11, p. 2673-2681, 1997.

Sutskever I., Martens J., Hinton G. E., « Generating text with recurrent neural networks », *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, p. 1017-1024, 2011.

Sutskever I., Vinyals O., Le Q. V., « Sequence to sequence learning with neural networks », *Advances in neural information processing systems*, p. 3104-3112, 2014.

Vilnis L., McCallum A., « Word representations via gaussian embedding », *arXiv preprint arXiv:1412.6623*, 2014.

Weston J., Bordes A., Yakhnenko O., Usunier N., « Connecting Language and Knowledge Bases with Embedding Models for Relation Extraction », *Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, p. 1366-1371, 2013.

Zaremba W., « An empirical exploration of recurrent network architectures », 2015.