Adrien Burq
Alexandre Giuly
Tuteur : François Yvon

Septembre - Décembre 2021

GÉNÉRATION DE PHRASES MULTILINGUES

RÉSUMÉ

La mondialisation qui régit les échanges au sein de notre monde moderne a modifié considérablement le vocabulaire que nous employons. Le langage courant s'est vu enrichi de plusieurs anglicismes qui ont été soumis aux règles grammaticales du français. Par exemple, "je suis overbooké", "ça boostera sa confiance", etc. Néanmoins, tout mélange anglais-français ne suffit pas à produire cette structure particulière. C'est dans ce contexte que nous nous proposons d'étudier comment générer de telles phrases.

Afin d'obtenir un modèle apte à répondre à cette problématique, nous nous pencherons sur l'utilisation des réseaux récurrents dans les Variational Auto-Encoder (VAE). Ces structures sont couramment employées en Natural Language Processing (NLP) afin d'effectuer des tâches de traduction mais surtout de génération de texte ou encore dans le but d'extraire le sens d'un extrait. C'est pourquoi nous décidons de les utiliser dans le cadre de notre sujet.

Nous tenons à remercier François Yvon pour la pédagogie et la disponibilité dont il a fait preuve au cours de cet Enseignement d'Approfondissement dédié à l'initiation à la recherche dans le domaine du NLP. Ses conseils et ses méthodes nous on permis d'appréhender notre sujet avec les bases nécessaires à sa compréhension. Nous avons eu l'opportunité de choisir parmi différents sujets en lien avec ses travaux de recherche ce qui nous a permis de découvrir avec motivation les enjeux du NLP.

SOMMAIRE

1
CONTEXTE THÉORIQUE

1.1 Réseaux récurrents et LSTM

1.1.1 • Théorie

Le principe des réseaux récurrents repose sur un mécanisme de compréhension propre à l'espèce humaine. En effet, dans une phrase, le sens du mot peut varier selon un contexte et dépend très fortement des mots qui le précèdent. Chaque mot n'est pas analysé indépendemment mais la pensée suit une ligne directrice à la lecture d'une phrase.

Les réseaux de neurones feed-forward ne sont pas capables de reproduire ce phénomène, en raison de la nécessité de gérer des phénomènes à distance non bornée, ce qui peut peut représenter un obstacle pour répondre à certains problèmes à l'instar de l'analyse des séries temporelles ou de la reconnaissance d'écriture manuscrite. Les réseaux récurrents, en introduisant une architecture récursive, permettent à l'information de subsister. Ce modèle peut être également être perçu comme la disposition successive d'un même module qui transmet de copies en copies une information appelée hidden state. Cette vision met en évidence un des principaux avantages des réseaux récurrents, l'input donné à un réseau peut être de taille variable, comme par exemple dans le cadre de la traduction automatique.

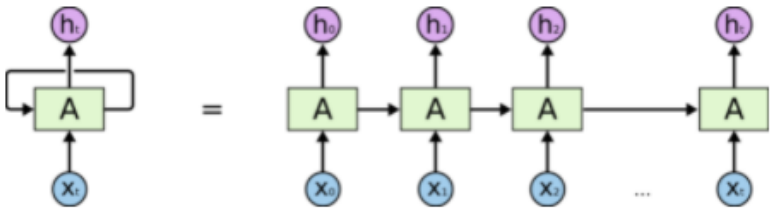


Figure 1: Architecture des RNNs

Néanmoins, la littérature (Hochreiter(1991) et Bengio(1994)) et les expériences mettent en évidence un problème de dépendance sur le long terme, inhérent aux réseaux récurrents classiques. Dans le cadre de la prédiction de mots dans une phrase, l'information pertinente à aller chercher n'est pas toujours à la même distance. Par exemple, dans la phrase "le chat boit du lait", la donnée de "chat" et "boit" permet de donner immédiatement le mot "lait". Cependant, dans "il joue très bien au foot [...], il rêve de devenir footballeur", l'acquisition de "foot" permet de prédire le métier correspondant "footballeur" mais la distance entre ces deux mots nuit à la transmission de cette donnée. Plus l'écart entre l'information et le mot à prédire est important, plus les réseaux récurrents ne seront pas aptes à effectuer leur tâche. L'introduction d'un nouveau type de réseaux récurrents par

Hochreiter Schmidhuber (1997), Long Short Term Memory (LSTM) permet de palier ce problème.

Les changements apportés à l'architecture classique concernent le module répété. Alors qu'il n'était constitué que d'un seul réseau de neurones, il en possède désormais quatre différents et inter-connectés. De plus, une variable supplémentaire est retournée en sortie de ce module, appelée cell state et constitue la clé de cette nouvelle solution. De module en module, elle va transmettre l'information à garder et son contenu est modifié en fonction des différents inputs et hidden states.[5]

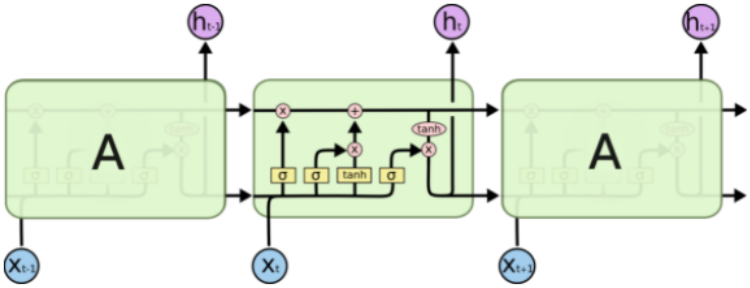
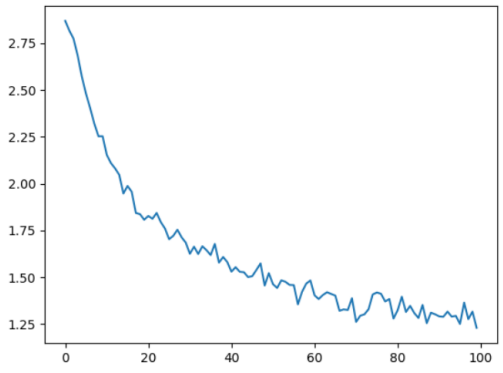


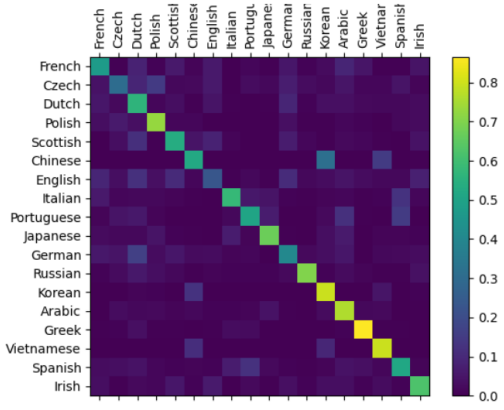
Figure 2: Architecture des LSTMs

1.1.2 • Application

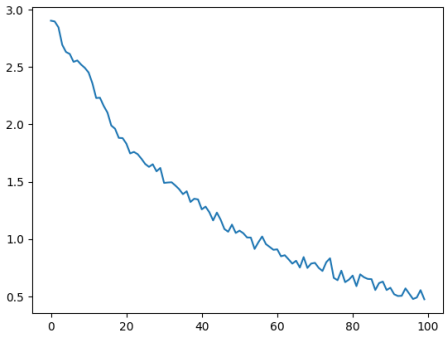
Afin de prendre en main les LSTMs, nous avons étudié un problème de classification de noms de famille, en fonction de leur écriture, entre 18 catégories de pays ou origines. Chaque nom est considérée comme une séquence de lettres qui peut être donnée à notre réseau récurrent. Une solution existant déjà pour les réseaux récurrents classiques (RNNs)[7], nous avons implémenté une solution qui utilise des LSTMs. Nous présentons ci-dessous la loss moyenne et la matrice des corrélations pour le RNN et pour le LSTM.



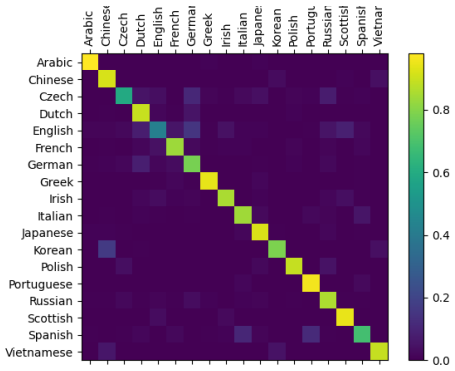
(a) Loss moyenne - RNN



(b) Matrice des corrélations - RNN



(a) Loss moyenne - LSTM



(b) Matrice des corrélations - LSTM

On constate que le LSTM fournit de meilleurs résultats.

1.2 Principe de l'auto-encodeur variationnel

Un auto-encodeur est constitué d'une fonction encodeur ϕ_{enc} et d'un modèle de décodeur probabiliste $p_\theta(x|z = \phi_{enc}(x))$. L'auto-encodeur maximise la vraisemblance d'obtenir x conditionné à l'encodage z de x [1]. Dans notre cas, les x sont les mots d'une langue et les z sont leur représentation dans l'espace latent. Pour un Auto-Encodeur quelconque, il n'y a aucune garantie sur les propriétés de l'espace latent. Un Auto-Encodeur Variationnel (VAE) assure que l'espace latent est continu. Cet espace est paramétré et le modèle apprend ces paramètres sur un échantillon d'entraînement. Puis, en utilisant la propriété de continuité, on est capable de générer des vecteurs proches des vecteurs d'entraînements dans l'espace latent.

Notre problème de génération de phrases est le suivant : à partir d'un dataset de texte, est-il possible de générer des phrases syntaxiquement correctes ayant un sens ? Pour cela, nous admettons qu'il est possible de définir un espace latent Z continu, paramétré et conditionné par notre base de donnée initiale, à partir duquel on peut générer des phrases à l'aide d'une distribution de probabilité $p_\theta(x|z)$ où l'on a représenté les mots sous formes de vecteurs x dans un espace de grande dimension. Le but est donc d'apprendre la distribution p_θ. Ainsi, on pourrait générer une phrase en tirant des z consécutivement et en prenant $x = argmax_x p_\theta(x|z)$. Le problème vient du fait que trouver la loi $p_\theta(z|x)$ est malheureusement souvent insoluble car par la loi de Bayes, on a :

$$p_\theta(z|x) = \frac{p_\theta(x|z)p_\theta(z)}{p_\theta(x)}$$

où le terme au dénominateur est inconnu. C'est pourquoi on introduit une distribution $q_\phi(z|x)$ qui est une approximation de $p_\theta(z|x)$ ce qui nous permettra d'apprendre conjointement les deux distributions $q_\phi(z|x)$ et $p_\theta(x|z)$.

Notre problème se découpe donc en deux parties : utiliser notre dataset pour caractériser l'espace latent à l'aide de la distribution $q_\phi(z|x)$, et retranscrire les tirages en phrases à

l'aide de la distribution $p_\theta(x|z)$. Ces deux parties sont interdépendantes et correspondent à l'encodage et au décodage de l'Auto-Encodeur Variationnel. Afin de trouver ces paramètres θ et ϕ, tout l'enjeu est de maximiser la log-vraisemblance de la distribution $p_\theta(x)$ sur notre jeu de données.

En suivant les calculs décrits dans [2] et [3], une solution serait de maximiser la probabilité de retrouver x en sortie du décodeur $p_\theta(x|z)$ ainsi que de minimiser la divergence de Kullback Leibler entre la distribution a postériori $q_\phi(z|x)$ et la probabilité à priori $p_\theta(z)$. En effet, la fonction

$$ELBO = \mathbb{E}_X[\mathbb{E}_{z\sim q_\phi(z|x)}[log(p_\theta(x|z)] - D_{KL}(q_\phi(z|x)||p_\theta(z))]$$

est une borne inférieur de la log-likelihood que l'on souhaite maximiser. Le premier terme représente l'erreur de reconstruction : pour un mot donné, dans quelle mesure est ce que notre VAE est capable de le reproduire fidèlement en sortie. Le second terme est la mesure de l'écart entre la distribution des mots apprise par le VAE sur le dataset et la distribution réelle. On suppose en général que la distribution de probabilité à priori sur l'espace latent est normale de paramètres $(\mu(X), \Sigma(X))$ que l'on ajuste au cours de l'apprentissage.

Cependant la back-propagation de l'erreur est difficilement réalisable lorsque l'on garde l'aléa dans la boucle. Ainsi, le "reparametrization trick" consiste à ne tirer que des vecteurs aléatoires selon la loi normale standard. On a :

$$\mathcal{N}(\mu(X), \Sigma(X)) = \mu + \Sigma^{\frac{1}{2}}\mathcal{N}(0, \mathbf{I})$$

Ainsi, l'estimation de la perte par une méthode Monte Carlo (basée sur des échantillons aléatoires) devient différentiable en μ et Σ. Cela permet de réaliser la back-propagation et d'ajuster les paramètres au cours de l'apprentissage comme montré sur la Figure 5.

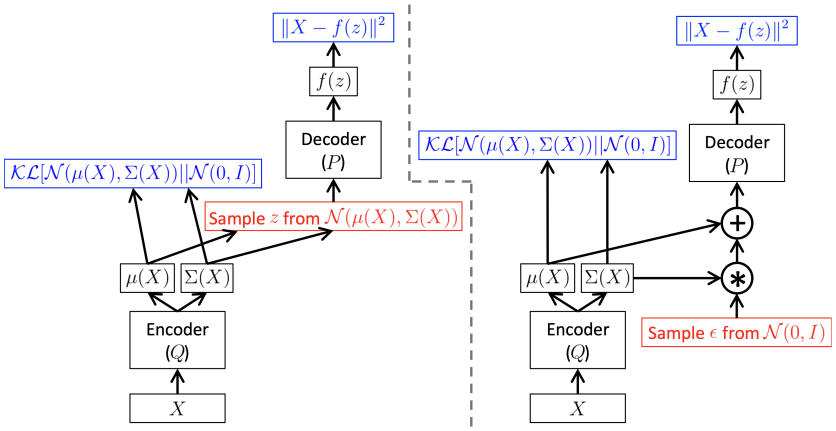


Figure 5: Principe de l'Auto-Encodeur Variationnel. A gauche sans le "reparametrization trick" et à droite avec [2].

Une fois que les paramètres $(\mu(X), \Sigma(X))$ sont appris, il est possible de générer des phrases en tirant des échantillons selon la loi $\mathcal{N}(0, \mathbf{I})$ et en utilisant ensuite le décodeur.

L'intérêt des réseaux récurrents vus dans la partie précédente est que la fonction encodeur se base sur un réseau récurrent pour générer une variable latente z prenant en compte le reste de la phrase. De même, le décodeur s'appuie sur un réseau récurrent pour générer des mots en fonction des mots déjà générés et ainsi produire une phrase syntaxiquement correcte (si l'entraînement s'est bien déroulé). Le principe est récapitulé à la Figure 6.

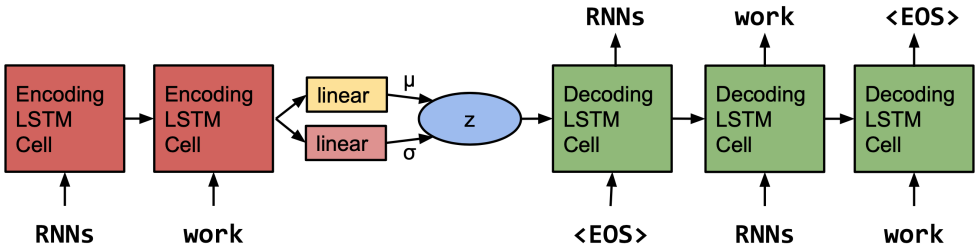


Figure 6: Exemple de structure de VAE utilisant des réseaux récurrents. Les mots sont représentés à partir d'un dictionnaire sous forme de vecteurs à l'aide d'un embedding [1].

1.3 Posterior-collapse

1.3.1 • Énoncé du problème

Dans le cadre d'un modèle de langue profond, les VAEs sont utilisés pour générer des phrases à partir d'un tirage sur un espace latent continu [1]. Un tel schéma, de part la facilité à travailler avec des distributions, donne lieu à de nombreuses pistes de travail telle que l'étude du voisinage de la variable latente d'une phrase. Néanmoins, les VAEs qui utilisent des RNNs présentent le défaut de souvent ignorer cette variable latente au cours de leur apprentissage. Le décodeur apprend à ignorer la variable latente et l'encodeur ne parvient pas à encoder de l'information. Des phrases dont les variables latentes sont dans le même voisinage peuvent ne pas avoir de lien sémantique ou de corrélation, ce qui reviendrai à simplement utiliser un AE. Ce problème est nommé posterior collapse.

Nous avons étudié plus haut que l'enjeu du problème était d'approcher la true posterior $p_\theta(z|x)$ par une approximation $q_\phi(z|x)$. Ce problème se ramène à optimiser l'ELBO selon θ et ϕ conjointement. Néanmoins, Si aucune information n'est codée dans la variable latente, celle-ci devient indépendante des data.
Le décodeur (θ) apprend alors à ignorer la variable latente et l'encodeur (ϕ) échoue dans sa tâche. Dans ce cas, le terme de reconstruction $\mathbb{E}_{z\sim q_\phi(z|x)}[log(p_\theta(x|z)]$ ne dépend plus que de θ. Par ailleurs le terme de divergence $D_{KL}(q_\phi(z|x)||p(z))$ ne dépend que de ϕ.
L'optimisation jointe selon θ et ϕ devient alors une optimisation selon θ pour la reconstruction et une optimisation selon ϕ pour D_{KL}. Cette divergence est alors trivialement réduite à zéro et la true posterior $p_\theta(x|z)$ devient égale à la prior $p(z)$.

1.3.2 • État de l'art des solutions

De nombreuses méthodes ont été développées pour répondre à ce problème [1][6].

KL Annealing Cette méthode consiste à introduire un poids variable devant le terme de divergence D_{KL} qui augmente progressivement de 0 à 1. De cette façon, le programme d'optimisation peut consacrer sa tâche à la maximisation de terme de reconstruction $\mathbb{E}_{z\sim q_\phi(z|x)}[log(p_\theta(x|z)]$ avant d'optimiser progressivement le terme de divergence.

Cyclic Annealing Il s'agit d'une variante de la méthode précédente. Le poids affectant le terme de divergence ne varie plus de façon monotone mais plutôt cycliquement.

Word dropout and historyless decoding Afin d'éviter que le décodeur en tant que tel ne performe sans faire appel aux variables latentes, il est possible d'affaiblir le décodeur. Une façon naturelle de procéder est de remplacer certains mots précédant l'observation par un mot générique inconnu. Une telle approche force le décodeur à se reposer sur la variable latente pour effectuer une bonne prédiction. Cette technique fait également à un paramètre $k \in [0, 1]$ pour lequel $k = 0$ correspond au cas où le décodeur ne voit aucun input et $k = 1$ celui où on ne cache aucun mot.

KL Thresholding/Free Bits (FB) La solution FB consiste à plafonner le terme de divergence par un seuil en dotant le D_{KL} d'un $max(\lambda, .)$. Cette fonction peut être appliquée à chaque coordonnée de la variable latente de la façon suivante :

$$\sum_i max[\lambda, D_{KL}(q_\phi(z_i|x)||p(z_i))]$$

λ est appelée target rate. Cette méthode oblige le programme d'optimisation à stopper la minimisation de la divergence pour les dimensions dont la coordonnée est plus petite que le target rate. Une autre application serait d'appliquer directement le max à l'ensemble de la divergence en écrivant :

$$max[\lambda, D_{KL}(q_\phi(z|x)||p(z))]$$

1.3.3 • Méthode retenue

Dans leur article [4], Bohan Li et Al. ont décidé d'entraîner dans un premier temps un simple auto-encodeur. Cela revient à utiliser une méthode de KL Annealing en mettant un poids de zéro à la divergence de Kullback-Leibler pendant l'entraînement du VAE. Ils entraînent ensuite une nouvelle fois leur VAE en utilisant une approche Free Bits et le VAE est ainsi entraîné avec l'ELBO complet. Cependant, un pré-entraînement seul ne suffit pas à éviter le "posterior collapse". Ainsi, il convient de coupler le pré-entraînement avec une méthode de KL Annealing pour obtenir de meilleurs résultats comme montré dans [4]. Le schéma retenu est simplement d'augmenter linéairement le poids sur la divergence de Kullback-Leibler pour passer de 0 à 1 au cours des 10 premières epochs tout en maintenant l'objectif de seuil avec les Free Bits. La littérature met en évidence que cette approche fournit de meilleurs résultats qu'un entraînement direct du VAE.

2
MODÈLE MULTILINGUE NAÏF

2.1 Génération multilingue et homotopies

La mondialisation qui régit les échanges au sein de notre monde moderne a modifié considérablement le vocabulaire que nous employons. Le langage courant s'est vu enrichi de plusieurs anglicismes qui ont été soumis aux règles grammaticales du français. Par exemple, "je suis overbooké", "ça boostera sa confiance", etc. C'est dans ce contexte que nous étudions comment générer de telles phrases.

Ainsi, notre but était de générer des phrases mélangeant plusieurs langues. Nous nous sommes intéressé au cas de l'anglais et du français. Pour cela, on utilise la méthodologie introduite précédemment en utilisant la continuité de l'espace latent.

L'entraînement d'un modèle suivant la méthodologie de l'article [4] fournit deux fonctions encodeur et décodeur : e et f. La première permet d'envoyer une phrase dans l'espace latent et la seconde permet de décoder un vecteur de l'espace latent en une phrase. En encodant deux phrases distinctes x_1 et x_2, on obtient deux vecteurs $e(x_1) = z_1$ et $e(x_2) = z_2$ dans l'espace latent. Comme cet espace est continu, on peut considérer un vecteur z comme combinaison convexe de z_1 et z_2. Ainsi, il est possible d'observer l'évolution des phrases $f(z)$ lorsque z décrit le segment $[z_1, z_2]$: $z = (1-\lambda)z_1 + \lambda z_2$ avec $\lambda \in [0, 1]$. Si l'apprentissage du VAE s'est effectué correctement, nous devrions retrouver $f(z_1) = x_1$ et $f(z_2) = x_2$. Pour λ proche de 0, nous nous attendons à retrouver des phrases $f(z)$ proches de $f(z_1)$ et pour λ proche de 1, nous espérons retrouver des phrases $f(z)$ proches de $f(z_2)$. Une telle approche permettrait ainsi de trouver des combinaisons des deux phrases de départ dont le sens varie peu à peu de l'une à l'autre. On désigne par le terme d'homotopie ces phrases interpolées.

Nous nous sommes appuyés sur cette technique pour générer du texte multilingue : nous prenons une phrase en français pour x_1 ainsqi qu'une phrase en anglais pour x_2. En faisant varier λ dans l'intervalle $[0, 1]$, on s'attend à générer un mélange des deux phrases, et donc un mélange de français et d'anglais. De plus, pour λ proche de 0, nous espérons trouver des phrases $f(z)$ en français majoritairement avec quelques mots d'anglais et pour λ proche de 1, nous devrions trouver des phrases $f(z)$ en anglais majoritairement avec quelques mots de français.

2.2 Premier modèle

2.2.1 • Preprocessing

Les textes sur lesquels nous avons travaillé sont issus de la base de donnée Tateoba. Nous avons utilisé un dataset qui est scindé en deux avec, d'une part des phrases en français et d'autre part, les traductions correspondantes en anglais. Afin d'exploiter ces données, nous les avons préalablement mises en minuscule avant de convertir ses caractères en Ascii dans un soucis d'uniformité et de simplification du modèle.

Nous avons ensuite concaténé ces deux datasets en un seul et mélangé les lignes du texte afin que notre réseau ne fasse pas de différence entre apprendre de l'anglais et du français. A partir de ce fichier pré-traité, nous avons extrait un dataset d'apprentissage ainsi qu'un dataset de test et d'évaluation dont les lignes comportaient plus de 50 caractères. Chacun de ces dataset dispose de 10 000 lignes.

2.2.2 • Méthode d'apprentissage

AE Dans un premier temps, nous entraînons notre modèle, par batch, en gardant un poids nul devant le terme de divergence $D_{KL}(q_\phi(z_i|x)||p(z_i))$. L'ELBO n'est alors plus qu'un terme de reconstruction. Nous avons vu qu'un tel procédé était équivalent à entraîner un auto-encoder. L'objectif de cette phase est de disposer d'un encodeur pré-entraîné qui puisse ensuite générer des phrases, sans se préoccuper d'une éventuelle unité sémantique pour les phrases disposant de variables latentes proches.

VAE Dans un second temps, nous chargeons les poids préalablement obtenus en réinitialisant ceux du décodeur afin d'écarter tout problème se rattachant au posterior collapse. Comme évoqué précédemment, nous commençons l'entraînement en adoptant la méthode Free Bits, avec un target rate ayant pour valeur 2, couplée à un KL Annealing, c'est à dire une augmentation progressive du poids attribué au terme de divergence dans l'ELBO.

2.3 Résultats

Pour évaluer notre modèle, nous avons créé des homotopies en partant d'une phrase aléatoire et de sa traduction. Ci-dessous nous avons montré les résultats pour trois phrases de départ différentes et des pas de λ dans $\{\frac{1}{2}, \frac{1}{3}, \frac{1}{8}, \frac{1}{23}\}$. Nous avons également évalué pour chaque phrase sa langue ainsi que la certitude avec laquelle la langue est détecté avec le module Google Compact Language Detector.

```
j 'ai besoin d 'une enveloppe .
i need to blame well 's .
regarde-moi is skiing fantastic? .
je vais etre une faveur pour 'effarouche .
i need an envelope .
```

```
1  fr, 0.9755149483680725
2  en, 0.8446259498596191
3  yo, 0.14992977678775787
4  fr, 0.9999980926513672
5  en, 0.7767305970191956
```

Figure 7: Homotopies, n= 3

```
elle a repris son souffle .
i need to blame well .
il vit bizarre avec des bateau .
don 't try .
les blessures trient essaye qu 'elle peut plait .
je voir mon lunettes de pain .
she caught her breath .
```

```
1  fr, 0.990230917930603
2  en, 0.4922855496406555
3  fr, 0.9564734697341919
4  en, 0.9999961853027344
5  fr, 0.9993517398834229
6  fr, 1.0
7  en, 0.8708698153495789
```

Figure 8: Homotopies, n= 5

```
etes-vous homo ?
what , many people have no interest
who does he fantastic?
es-tu ami est demain ?
ouvrez la nuit !
where i should snow to believe of the bus?
do that help those this?
do you like a great smaller?
what are we happy to croissante in line?
who is a similar blood girls he may is here there became an main shorter
suitcase for the highwayman in a weather a unity of america .
if you want to care everything like for you?
are you gay?
```

```
1   fr, 0.9756898283958435
2   en, 0.9988048076629639
3   en, 0.9416285157203674
4   ca, 0.39330556988716125
5   fr, 0.9999523162841797
6   en, 0.9991286993026733
7   en, 0.9913515448570251
8   en, 0.9944934248924255
9   en, 0.998046875
10  en, 0.9999752044677734
11  en, 0.9998683929443359
12  en, 0.9999828338623047
```

Figure 9: Homotopies, n= 10

```
le ver de terre gigota lorsque je le touchai .
i need to blame well 's .
i drink him fantastic?
she were written glasses to my school is barking .
he was possible to tell you .
mon pere a contre-courant .
tom a-t-il a des anniversaire .
les phrases se cote l 'autre deja croissante toute un poupee et esprits
de livres pour la cuisine montant et mois pour le 'urgent .
en indisponible maintenant j 'etais attaquaient en talent .
il a accepte ce qu 'il soit un bon political hatives .
je ne 'en 'adressez pas pourrais-je please! .
danced utiliser les rangea en vide avec un regardent qui concerne .
je pense que nous ce vivants .
je va vraiment laisse un devrions demain .
je pense que c 'est un peu interessante .
gras je me fiez .
je ne voulais pas vraiment sac .
il s 'est tres menteur .
je suis un boulot .
folle ici than je vis .
c 'est le gauche qui etes-vous .
un rejette n 'a pas vu je retient .
elle se 'ont facile .
je n 'ai pas a l pitie de bruit , je l 'ai ete occupe .
il faut souvent devant son chambre .
ce n 'est pas son apres-midi .
the earthworm wriggled when i touched it .
```

Figure 10: Homotopies, n= 25

On observe tout d'abord que pour $\lambda = 0$ et $\lambda = 1$, les phrases retrouvées par le modèle sont différentes des phrases initiales. Par exemple, "J'ai besoin d'une enveloppe." ne devrait pas donner "I need to blame well's." après être passé dans l'encodeur puis dans le décodeur. Cela met en évidence que l'auto-encodeur n'est pas bien entraîné avant même de regarder si on peut retrouver des phrases syntaxiquement proches autour des phrases initiales.

D'autre part, Nos résultats mettent en évidence l'absence de structure de l'espace latent. En effet, on peut changer de langue plusieurs fois sur le segment entre les représentations dans l'espace latent d'une phrase et de sa traduction, et ce sans garder de similarités au niveau du sens des phrases.

Enfin, on remarque qu'il y a peu de mélange entre les deux langues : la majorité des phrases sont exclusivement dans une langue ou dans l'autre, comme on le vois avec la précision avec laquelle on peut évaluer les langues des différentes phrases. Cette dernière remarque peut être due à l'entraînement qui se fait exclusivement sur des phrases unilingues. Ainsi, les phrases générées par le réseau récurrent du décodeur sont entraînées pour être unilingues.

2.4 Limites du modèle

Une première limite que nous avons repérée est donc le risque d'avoir des représentations des deux langues totalement décorrélées dans l'espace latent (cf Figure 11). En effet, lorsque l'on entraîne le VAE, celui-ci s'entraîne alternativement sur des phrases anglaises ou françaises mais jamais les deux à la fois. Ainsi, une phrase et sa traduction peuvent se retrouver à deux endroits très éloignés de l'espace. Ainsi, une interpolation entre deux phrases qui sont la traduction l'une de l'autre peut donner des homotopies qui n'ont rien à voir avec les phrases initiales et ce même pour des λ proches de 0 ou de 1.

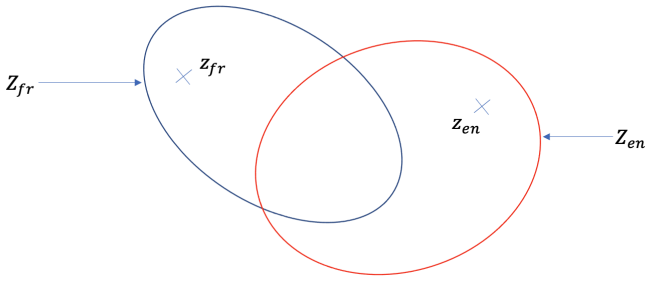


Figure 11: Schéma de la représentation du français et de l'anglais dans l'espace latent.

Une seconde limite peut venir du fait que l'apprentissage de deux langues différentes est plus long. En effet, les mots de chaque langue sont totalement décorrélés. En utilisant un

dataset de la même taille que pour une langue unique, on double la taille du vocabulaire. Ainsi, pour obtenir des résultats aussi bons dans chaque langue individuellement que lors de l'entraînement pour une langue unique, il faudrait peut-être avoir un dataset plus gros que pour une langue unique.

3 NOTRE MODÈLE

3.1 Modifications

3.1.1 • preprocessing

Dans notre nouveau modèle, nous nous proposons de rajouter un preprocessing supplémentaire en faisant appel à SentencePiece. Cette API permet de créer, à partir d'un dataset, un vocabulaire composé des mots ou morceaux de mots les plus fréquents. Il est ensuite possible d'écrire à nouveau nos jeux de données en utilisant seulement des mots de ce vocabulaire.

3.1.2 • Fonction de perte

Les résultats obtenus précédemment mettent en évidence que l'espace latent ne dispose pas d'une structure caractéristique. Les phrases en français et en anglais s'alternent sans qu'aucune structure ne soit visible. Dans la méthode précédente, nous n'avons exploité le lien qui existe entre une phrase en français et sa traduction en anglais.

C'est pourquoi, nous forçons cette connexion en rajoutant dans la fonction de perte un terme en $\frac{1}{2}||z_{fr} - z_{en}||^2$. En effet, étant donné que ces deux phrases partagent le même sens, nous souhaitons que leurs variables latentes soit proches. Un tel procédé permettrait de garantir une unité sémantique inter-langages.

3.1.3 • Architecture globale

Pour se faire, nous avons modifié notre méthode d'entraînement. Jusqu'à présent, des phrases étaient choisies au hasard indépendemment de la langue dans laquelle elles ont été écrites. Dans le cadre de notre modèle, nous avons créé des datasets d'entraînement, de test et d'évaluation pour chacune des langues en s'assurant que chaque ligne d'un dataset en français correspondait à sa traduction en anglais.

Au cours de l'apprentissage, pour chacune de ces lignes, notre réseau fournit une fonction de perte pour la version française $\mathcal{L}_{fr}$ et sa variable latente z_{fr} ainsi qu'une fonction de perte pour son homologue anglaise $\mathcal{L}_{en}$ accompagnée de sa variable latente z_{en}. Nous effectuons alors la back-propagation sur la loss finale :

$$\mathcal{L} = \frac{1}{2}(\mathcal{L}_{fr} + \mathcal{L}_{en}) + \frac{1}{2}||z_{fr} - z_{en}||^2$$

3.2 Résultats

3.2.1 • Méthodes de décodage

Dans le cadre de la génération de nos homotopies, nous avons testé différentes méthodes de décodage.

Méthode "greedy" Cette méthode est la plus intuitive et celle qui se fie le plus à la sortie du décodeur. Une couche linéaire à la suite du LSTM fournit une liste de logits qui attribue à chaque mot du vocabulaire un poids. Le mot qui possède l'argument le plus élevé est alors choisi. Néanmoins cette méthode fait aveuglement confiance à notre réseau, ce qui peut conduire à obtenir souvent le même résultat qui, de plus, n'est pas nécessairement optimal.

Méthode "sample" Afin d'atténuer ce phénomène, la méthode "sample" se propose de convertir la liste de logits en probabilités via un Softmax. Un tirage est ensuite réalisé suivant cette loi de probabilité, ce qui permet d'apporter un peu plus de diversité dans nos résultats.

Méthode "beam search" L'algorithme sélectionne à chaque passage dans le LSTM les K meilleures alternatives, i.e. les plus probables, en fonction de l'input précédent. Chacune des ces alternatives devient alors l'input d'un autre réseau et sur la totalité des résultats de sortie on garde à nouveau les K plus probables.

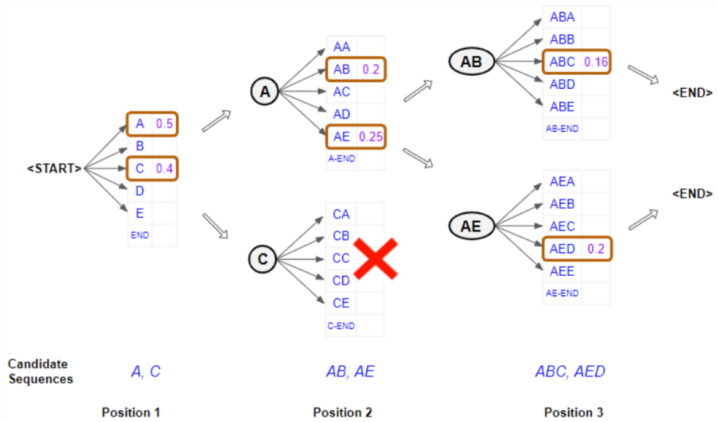


Figure 12: Schéma de la méthode "beam search"

Une telle approche permet d'obtenir une solution plus optimale que la méthode "greedy".

3.2.2 • Présentation des résultats

```
_nous _eu mes _de _nombreuse s _experience s _am er es _ durant _la _guerre
_j _' _looks _tous
_il _quel _mis _que _d
_quel es _es _mis eur _un oir
_je _semble m _n _' _mis _mis _m _wish _train
_j _' _mis _j _m eur _m
_zoo _ne ait _mis
er _son eur
_j _' est _cuisine ambi _puisse
_hotel _est _moin
_il cette _n _' vous _can _care ink _m _chez _c _m _prudent
_we _had _many _bit ter _experience s _dur ing _the _war
```

Figure 13: Homotopies en utilisant SentencePiece et la méthode ”sample”

```
nous  eu mes  de  nombreuse s  experience s  am er es  durant  la  guerre
j ' looks tous
il quel mis que d
queles es miseur unoir
je semblem n ' mis mis m wish train
j ' mis j meur m
zoo neait mis
er soneur
j 'est cuisineambi puisse
hotel est moin
ilcette n 'vous can careink m chez c m prudent
we  had  many  bit ter  experience s  dur ing  the  war
```

Figure 14: Homotopies en utilisant SentencePiece après décodage

Lors de l'utilisation de SentencePiece, nous avons découpé notre base de données en un vocabulaire de 3000 mots. On remarque que les mots formés par le VAE ne sont parfois ni du français, ni de l'anglais. Cela provient sans doute du fait que nous avons utilisé un petit vocabulaire. Nous pourrions ré-entraîner notre modèle avec un vocabulaire d'au moins 8000 mots. En effet, plus la taille du vocabulaire est petite et plus les mots seront hachés dans la mesure où ils engendreront une séquence de petits mots du vocabulaire. Le plus petit dictionnaire possible est constitué uniquement des 26 lettres de l'alphabet. Ainsi, lorsque les mots sont recombinés à partir de ce vocabulaire, on risque de former des mots qui n'existent pas.

Nous ne rencontrons pas ce problème avec la méthode de création de vocabulaire intégré dans le travail de Bohan Li et Al. [4]. En effet, ils constituent le vocabulaire à partir de tous les mots présents dans la base de textes donnée pour l'entraînement. La contrepartie de cette méthode est que le vocabulaire créé peut devenir très gros, ce qui peut augmenter les temps de calcul de l'encodeur et du décodeur, et par conséquent le temps d'entraînement

du modèle. Le choix du nombre de mots dans le vocabulaire est un compromis entre le nombre de choix possibles et le nombre de passages dans le réseau récurrent : pour un petit vocabulaire, une phrase est découpée en de nombreux petits morceaux qui doivent passer un par un dans le réseau. En sortie cependant, le décodeur doit choisir parmi une petite famille de mots possibles. A l'inverse, pour un gros vocabulaire, la phrase est peu découpée mais le nombre de choix possible pour le décodeur est plus grand.

Dans notre cas, l'utilisation de SentencePiece avec un vocabulaire de 3000 mots n'a pas considérablement amélioré les résultats et le temps de calcul. Puisque nous travaillons avec des jeux de données assez petits, nous avons utilisé la méthode de l'article [4] pour former le vocabulaire et comparer les différentes méthodes de décodage.

```
je vous crois totalement .
je voudrais le pluie .
ils devons proscrit que tu animaux parle .
je sais que vous ne y pas du lit .
je y pouvons des amis .
je n 'ai pas envie .
je pense que demain .
je suis un across? probleme .
je vois que c 'etait un nouveau .
c 'est a committee temoigna ceci courrier .
tom se jeux l 'amour de toutes cette amie .
i totally believe you .
```

```
fr, 0.9969110488891602
fr, 0.9999961853027344
fr, 0.9421082139015198
fr, 1.0
fr, 0.9968103170394897
fr, 0.849618673324585
fr, 0.8859052658081055
fr, 0.9974930882453918
fr, 1.0
la, 0.6247977018356323
fr, 0.9999980926513672
en, 0.9825230240821838
```

Figure 15: Homotopies avec la méthode "sample"

```
soyez plus gentils avec ton frere !
<s> je n 'ai pas envie .
<s> je n 'ai pas envie .
<s> je n 'ai pas envie .
<s> je n 'ai pas envie .
<s> je n 'ai pas besoin .
<s> je n 'ai pas vu !
<s> qu 'est-ce que je veux ?
<s> qu 'est-ce que je veux ?
<s> qu 'est-ce que je veux ?
<s> qu 'est-ce que je veux ?
be nicer to your brother .
```

```
fr, 0.8560422658920288
fr, 0.9696584939956665
fr, 0.9696584939956665
fr, 0.9696584939956665
fr, 0.9696584939956665
fr, 0.7073825001716614
fr, 0.9952486753463745
fr, 1.0
fr, 1.0
fr, 1.0
fr, 1.0
en, 0.9989134073257446
```

Figure 16: Homotopies avec la méthode "beam"

En multipliant les essais pour chaque méthode, nous avons validé les points exposés à la partie précédente : la méthode "sample est probabiliste puisqu'avec une même phrase de départ, on obtient des phrases interpolées différentes à chaque essai, contrairement aux méthodes "beam search" et "greedy" qui fournissent toujours les mêmes résultats. Les méthodes "beam et "greedy" fournissent des phrases interpolées plus proches les unes des

```
la jeune girafe suivait le troupeau .
je n 'ai pas le faire .
je n 'ai pas le faire .
je n 'ai pas le faire .
je n 'ai pas vu de l 'argent .
je n 'ai pas vu a la maison .
je n 'ai pas vu a la maison .
je n 'ai pas vu a la maison .
je n 'ai pas vu a la maison .
je n 'ai pas vu a la maison .
je n 'ai pas vu a la maison .
the young giraffe was following the herd .
```

```
fr, 0.4981188476085663
fr, 0.9997482895851135
fr, 0.9997482895851135
fr, 0.9997482895851135
fr, 0.9999542236328125
fr, 0.9991897344589233
fr, 0.9991897344589233
fr, 0.9991897344589233
fr, 0.9991897344589233
fr, 0.9991897344589233
fr, 0.9991897344589233
en, 0.5352619290351868
```

Figure 17: Homotopies avec la méthode ”greedy”

autres syntaxiquement et lexicalement, mais peuvent être très éloignées des phrases de départ.

Les méthodes ”beam” et ”greedy” mettent en évidence que l'espace latent a une certaine structure. En effet, le sens des phrases interpolées varie peu et les phrases de départ sont proches de leur traduction dans l'espace latent. ”Je n'ai pas le faire.” est proche de ”Je n'ai pas vu la maison.” par exemple. Cependant, on remarque également que l'Auto-Encodeur est très mauvais puisque une phrase comme ”La jeune girafe suivait le troupeau.” devient ”Je n'ai pas le faire” après passage dans l'encodeur et dans le décodeur.

Enfin, on remarque que les mélanges entre les deux langues apparaissent plus souvent lorsque l'on utilise la méthode ”sample”. Nous privilégions donc cette méthode pour faire de la génération multilingue. Cependant, la structure grammaticale des phrase n'est pas encore bien conservée par le modèle.

Ainsi, notre nouvelle méthode permet bien de structurer l'espace latent en rapprochant les représentations d'une phrase et de sa traduction. Cela n'améliore pas cependant l'Auto-Encodeur lui-même.

CONCLUSION

A travers cet Enseignement d'Approfondissement, nous avons pu nous initier à la recherche avec l'étude de plusieurs articles sur l'état de l'art des modèles génératifs en traitement du langage ainsi qu'avec un projet que nous avons mené au cours de la deuxième partie de l'enseignement. Nous avons pris en main les techniques d'Auto-Encodeur Variationnel développées dans les articles que nous avons étudiés et nous nous sommes particulièrement appuyés sur les travaux Bohan Li et al [4] qui ont élaboré un modèle de VAE pour la génération de phrases unilingue. Nous avons réussi à adapter ce modèle pour la génération multilingue mais le modèle nécessite encore quelques amélioration. Voici quelques perspectives que nous pourrions envisager dans cette optique.

Afin de générer des anglicismes, nous avons travaillé sur des données en anglais et en français en forçant l'apparition de ces mélanges par l'exploitation de la continuité de l'espace latent. Néanmoins, les structures que nous voulons générer ne naissent pas de la simple alternance de mots anglais et français. Les phrases multilingues sont difficilement créées par ces interpolations qui ne constituent qu'une première approche du problème. En effet, l'objectif n'est pas d'obtenir n'importe quel mot anglais qui "pop" au milieu d'une phrase française mais bien de générer cet agencement grammatical, dans la mesure où certains mots d'une langue ne sont jamais utilisés dans l'autre. Une piste d'amélioration serait alors d'utiliser notre modèle et de relancer une phase d'entraînement sur un dataset de phrases en franglais présentant ces structures afin d'augmenter la probabilité que des mots anglais apparaissent aux bons endroits dans une phrase française.

De plus, les travaux de Bohan Li et al [4] mettent en évidence que l'ELBO n'est pas nécessairement une borne optimal de la log-vraisemblance. Ainsi, une maximisation de ce terme pourrait ne pas toujours conduire au résultat escompté, c'est à dire la maximisation de la vraisemblance.

REFERENCES

[1] Samuel R. Bowman et al. *Generating Sentences from a Continuous Space*. 2015. arXiv: 1511.06349.

[2] Carl Doersch. *Tutorial on Variational Autoencoders*. 2016. arXiv: 1606.05908.

[3] Diederik P Kingma and Max Welling. *Auto-Encoding Variational Bayes*. 2013. arXiv: 1312.6114.

[4] Bohan Li et al. *A Surprisingly Effective Fix for Deep Latent Variable Modeling of Text*. 2019. arXiv: 1909.00868.

[5] Christopher Olah. *Understanding LSTM Networks*. 2015. URL: https://colah.github.io/posts/2015-08-Understanding-LSTMs/.

[6] Tom Pelsmaeker and Wilker Aziz. *Effective Estimation of Deep Generative Language Models*. 2019. arXiv: 1904.08194.

[7] Sean Robertson. *NLP from scratch: classifying names with a character-level RNN*. 2021. URL: https://pytorch.org/tutorials/intermediate/char_rnn_classification_tutorial.html.