Sommaire

1 Les modèles de langues neuronaux

1.1 Cas général

1.1.1 Construction de l'espace probabilisé

Définition 1.1 *On appelle vocabulaire un ensemble fini quelconque, noté $V = \{s_1, \dots, s_{|V|}\}$. Les s_i sont appelés symboles. On note ε le symbole vide qui n'appartient pas à V.*

Exemple de symboles :

— Un caractère

— Un mot

— Un bit

Exemple de vocabulaire :

— Ensemble des mots de la langue française

— Ensemble des caractères unicode

Définition 1.2 *Un texte est un élément de V^L, où $L \in \mathrm{N}^*$.*

On cherche à définir une probabilité sur l'ensemble des textes. Définissons notre espace de probabilité.

Définition 1.3 *On appelle l'ensemble des textes $\Omega = \bigcup_{L=1}^{+\infty} V^L$. On note $\mathcal{A} = \sigma\left(\{\{T\} \| t \in \Omega\}\right)$, une tribu sur Ω.*

On note $L : T \in \Omega \mapsto |T|$ la variable aléatoire qui associe à un texte sa longueur. On définit les $(X_n)_{n\in\mathrm{N}^*}$ comme :

$$\forall i \in \mathrm{N}^*, X_i(T) = \begin{cases} i\text{-ème symbole de } T \text{ si } i \leqslant L(T) \\ \varepsilon \text{ si } i > L(T) \end{cases}$$

On suppose qu'il existe une probabilité $\mathbb{P}$ sur l'espace probabilisable $(\Omega, \mathcal{A})$. On dispose d'un échantillon de textes distribué selon la mesure $\mathbb{P}$ et on cherche à estimer $\mathbb{P}$ par une mesure de probabilité $\widehat{\mathbb{P}}$.

On appelle $\widehat{\mathbb{P}}$ un modèle de langue. En raison de la nature séquentielle du langage, on le construit en pratique en conditionnant sur les mots précédents du texte.

1.1.2 Construction d'un modèle de langue par probabilités conditionnelles

Soit un texte $T = s_1 \dots s_L \in V^L$, où $L \in \mathbb{N}^*$.
La probabilité d'observer T s'écrit :

$$\begin{aligned}
\mathbb{P}(T) &= \mathbb{P}\left(\bigcap_{i=1}^{L} X_i = s_i \cap \bigcap_{i=L+1}^{+\infty} X_i = \varepsilon\right) \\
&= \mathbb{P}\left(\bigcap_{i=1}^{L} X_i = s_i \cap X_{L+1} = \varepsilon\right) \text{ par construction des } X_i \\
&= \mathbb{P}(X_1 = s_1 \cap \ldots \cap X_L = s_L \cap X_{L+1} = \varepsilon) \\
&= \mathbb{P}(X_{L+1} = \varepsilon | X_1 = s_1, \ldots, X_L = s_L)\mathbb{P}(X_1 = s_1, \ldots, X_L = s_L) \\
&= \mathbb{P}(X_{L+1} = \varepsilon | X_1 = s_1, \ldots, X_L = s_L)\mathbb{P}(X_L = s_L | X_1 = s_1, \ldots, X_{L-1} = s_{L-1}) \times \\
&\qquad \mathbb{P}(X_1 = s_1, \ldots, X_{L-1} = s_{L-1}) \\
&= \prod_{i=1}^{L+1} \mathbb{P}(X_i = s_i | X_1 = s_1, \ldots, X_{i-1} = s_{i-1}) \text{ en posant } s_{L+1} = \varepsilon
\end{aligned}$$

Si la séquence de mots est longue, cela devient rapidement coûteux en mémoire (taille de type factoriel). En effet, il faut être capable d'estimer la probabilité $\mathbb{P}(m_i | m_1 \dots m_{i-1})$. Nous sommes donc amenés à effectuer des approximations dans les calculs pour estimer ces probabilités. Ces différentes estimations conduisent à la définition de différents modèles de langues neuronaux.

Nous distinguons les modèles de langue suivants :

— les modèles n-grams
— les modèles Neural Network (NN)

1.2 Modèle n-gram

Dans un modèle n-gram, nous supposons que la probabilité d'apparition du mot $m - i$ ne dépend que de $n - 1$ prédécesseurs.

— Cas $n = 1$: Modèle unigram : $\mathbb{P}(s) = \prod_{i=1}^{N} \mathbb{P}(m_i)$
— Cas $n = 2$: Modèle bigram : $\mathbb{P}(s) = \prod_{i=1}^{N} \mathbb{P}(m_i | m_{i-1})$
— Cas $n = 3$: Modèle trigram : $\mathbb{P}(s) = \prod_{i=1}^{N} \mathbb{P}(m_i | m_{i-2} m_{i-1})$
— Cas $n > 3$: Modèle n-gram : $\mathbb{P}(s) = \prod_{i=1}^{N} \mathbb{P}(m_i | m_{i-(n-1)} \dots m_{i-1})$

Il est cependant très difficile de calculer ces probabilités dans l'absolue. Nous estimons alors ces probabilités sur un corpus de textes et nous supposons que le corpus de textes reflète la langue dans l'absolue. Nous devons donc disposer d'un très grand corpus de textes. Il s'agit là d'une approche statistique.

Etant donné que nous travaillons sur un corpus de textes fini, nous utilisons naturellement pour probabilité la mesure de comptage. Ainsi, le calcul de probabilité conditionnelle devient :

$$\mathbb{P}(m_i | m_{i-(n-1)} \dots m_{i-1}) = \frac{|m_{i-(n-1)} \dots m_{i-1} m_i|}{|m_{i-(n-1)} \dots m_{i-1}|}$$

Limitations : Etant donné que nous travaillons sur un corpus fini, nous avons une combinaison de mots finis. Il est possible qu'un mot qui n'apparaît pas dans le modèle. Sa probabilité d'apparition est donc nulle : $\mathbb{P}(m_k) = 0$. On parle de sparcité. Cette probabilité nulle pose problème : toute séquence de mots qui contient un mot qui n'est pas dans le corpus a une probabilité égale à 0 d'apparaître. Notre modèle reconnaît donc uniquement des séquences connus.

Pour palier ce problème et pouvoir généraliser à des séquences de mots non connus, nous pouvons effectuer un « lissage », qui consiste à attribuer une valeur de probabilité non nulle pour les mots n'apparaissant jamais dans le corpus.

1.3 Modèle Neural Network

Les modèles n-gram utilisent une approche statistique pour estimer les probabilités d'apparition d'une séquence de mots. Pour qu'elle soit efficace, il est nécessaire d'avoir un grand corpus de textes. Une autre approche de représentation des probabilités est d'utiliser les réseaux de neurones. L'idée est de capturer les liens (ou caractéristiques) que les mots peuvent avoir entre eux via un réseau de neurones. Ces liens sont représentés par les différentes connexions qui existent entre les neurones du réseau. On parle de « représentation distribuée ». Nous passons alors d'une représentation discrète à une représentation continue à travers un vecteur. Chaque mot est représentée par un vecteur dans $\mathbb{R}^m$. Cette représentation continue permet de représenter énormément de combinaisons possibles en modifiant très légèrement le vecteur qui représente la séquence de mots.