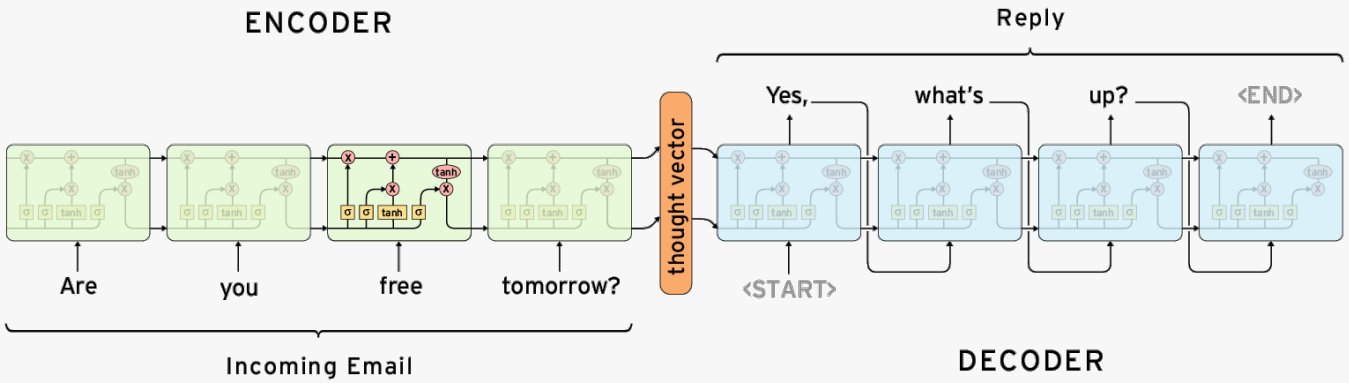


Comprendre le langage humain

Approches neuronales

Patrice Béchard
MILA

13 novembre 2018

Pourquoi le traitement des langues naturelles ?

- Énormément de données disponible en ligne sous forme textuelle.
- On veut faire plusieurs choses avec cette information :
  - Résumer pour trouver l'information importante
  - Traduire automatiquement d'une langue à une autre
  - Détecter le sentiment des utilisateurs d'un produit
  - Avoir une réponse à une question
  - etc.

C'est quoi le traitement des langues naturelles ?

- Branche de l'intelligence artificielle visant la compréhension du langage.
- Natural Language Processing/Understanding (NLP/NLU)
- On y compte des tâches telles que :
  - Résumés automatiques de texte
  - Traduction automatique
  - Analyse de sentiments
  - Systèmes de question et réponse
  - Agents conversationnels
  - Reconnaissance vocale
  - etc.

Plan de l'exposé

Bases de l'apprentissage automatique

Réseaux de neurones et apprentissage profond
- Modèles linéaires
- Perceptron multicouche
- Réseaux de neurones récurrents
- Modèles séquence à séquence

Bien représenter les mots
- Sac de mots
- Word embeddings

Tâches en NLP
- Analyse de sentiments
- Modèles de langue
- Traduction automatique
- Agents conversationnels

Bases de l'apprentissage automatique

Deux approches à l'intelligence artificielle

IA «classique» symbolique

- Fondé sur le raisonnement logique
- Règles codées à la main (`if... else...`)
- Pas de gestion de l'incertain

Apprentissage automatique

- Apprendre les paramètres à partir d'exemples
- Approche probabiliste
- Objectif : **généralisation**

Types de problèmes

- Apprentissage supervisé
  - Régression
  - Classification
- Apprentissage non-supervisé
  - Réduction de dimensionalité, clustering, détection d'anomalie, ...
- Apprentissage par renforcement

Apprentissage supervisé

- Ensemble de données
  $\mathcal{D}_{\text{train}} = \{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(N)}, y^{(N)})\}$.
- Le modèle prédit une sortie $f(x^{(i)})$ en fonction de l'entrée $x^{(i)}$. On veut que le modèle prédise la cible $y^{(i)}$.
- Lors de la phase d'entraînement, le modèle ajuste ses paramètres Θ dans le but de minimiser le risque empirique $\hat{R}$ selon une fonction de coût $\mathcal{L}(y, f(x))$

$$\hat{R} = \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}\left(y^{(i)}, f(x^{(i)})\right) \qquad \Theta^* = \arg\min_{\Theta} \hat{R}(f, \mathcal{D}_{train})$$

Quelques fonctions de coût...

- Erreur quadratique (régression) :

$$\mathcal{L}(y, f(x)) = ||y - f(x)||_2^2$$

- Erreur de classification :

$$\mathcal{L}(y, f(x)) = \mathbb{1}_{\{y \neq f(x)\}}$$

- Entropie croisée binaire :

$$\mathcal{L}(y, f(x)) = -y \log f(x) + (1 - y) \log(1 - f(x))$$

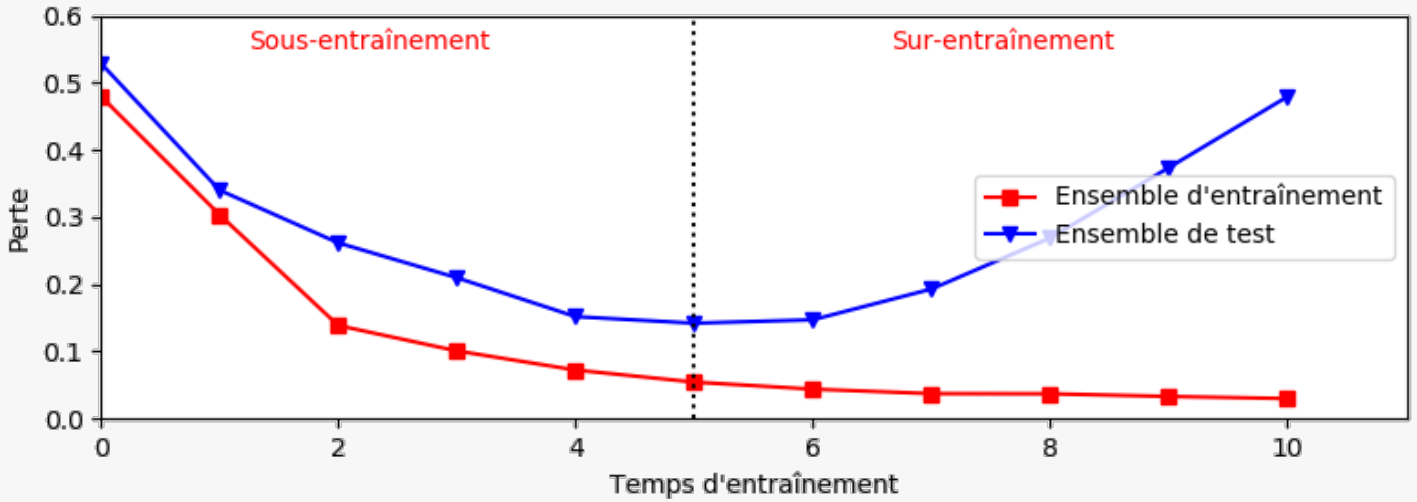
- Moins log-vraisemblance :

$$\mathcal{L}(y, f(x)) = -\log f(x)_y$$

Estimer l'erreur de généralisation

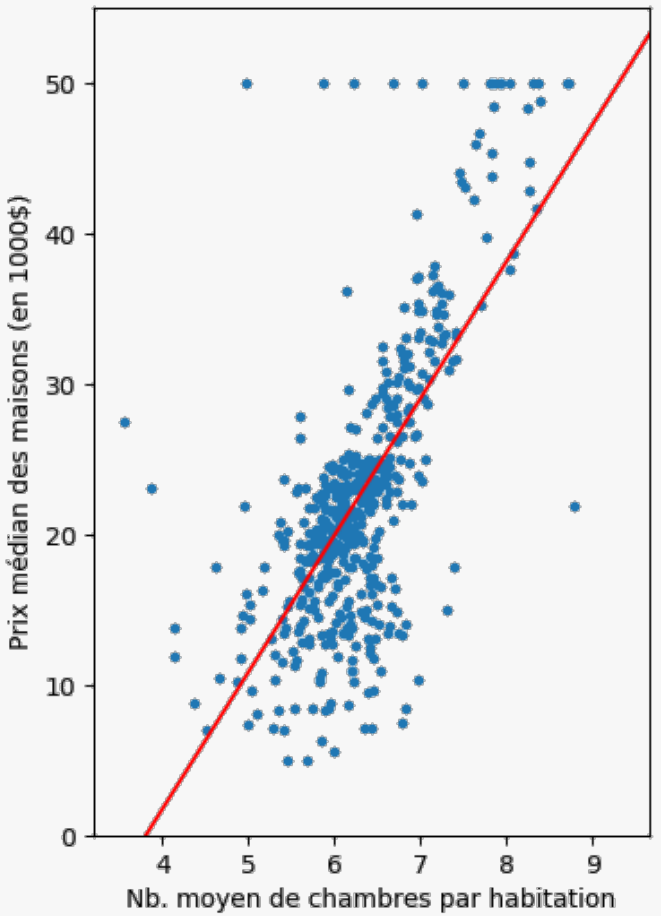
- But de l'apprentissage machine : **généralisation**
- On test le modèle entraîné sur des exemples jamais vus auparavant :

$$\mathcal{D}_{\text{test}} = \{(x^{(1')}, y^{(1')}), (x^{(2')}, y^{(2')}), \ldots, (x^{(N')}, y^{(N')})\}$$



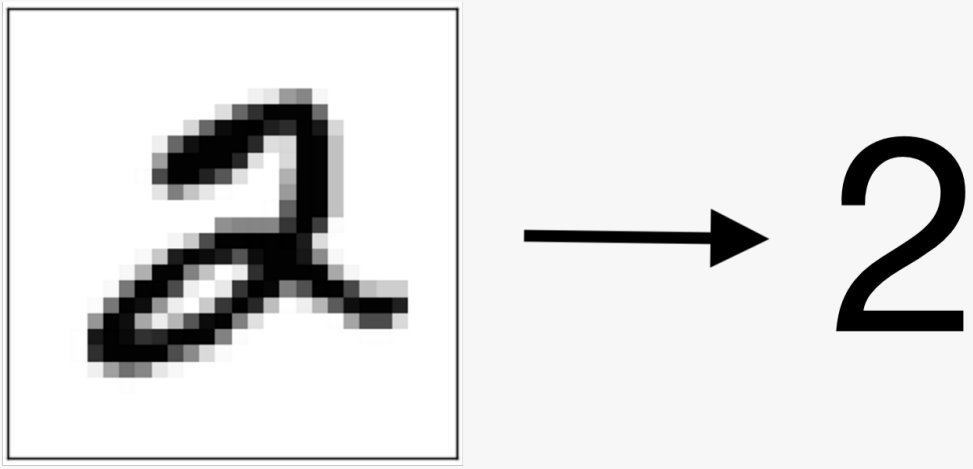
Exemple : Prédiction du prix des maisons à Boston [4]

- Problème de régression
- **Entrée** : 13 traits caractéristiques ($\in \mathbb{R}^{13}$)
  ex : Nb. de chambres, Taux de criminalité, ...
- **Sortie** : Prix de la maison ($\in \mathbb{R}$)



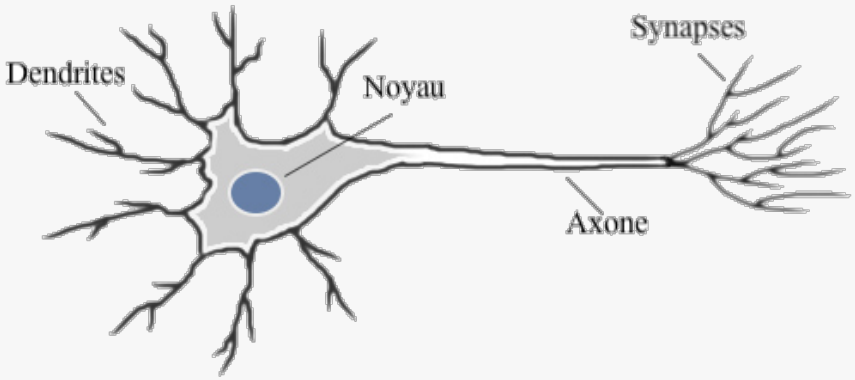
Exemple : Classification de chiffres manuscrits [7]

- Problème de classification
- Entrée : Image 28x28 px
- Sortie : Chiffre de 0 à 9

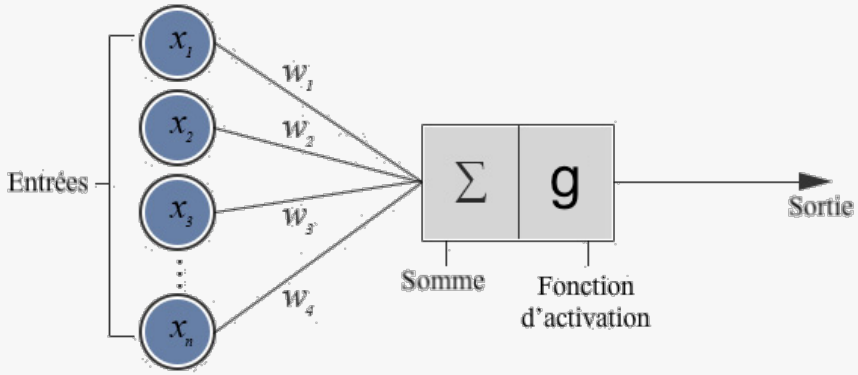


Réseaux de neurones et apprentissage profond

Neurone biologique

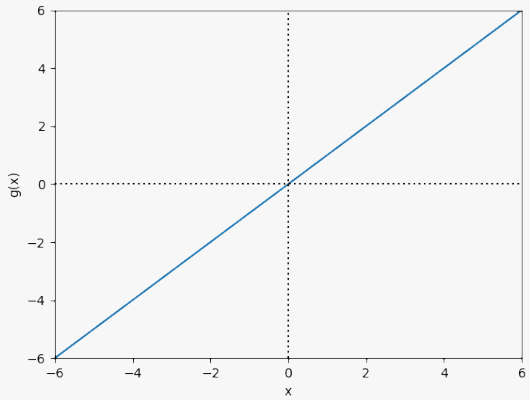


Neurone artificiel

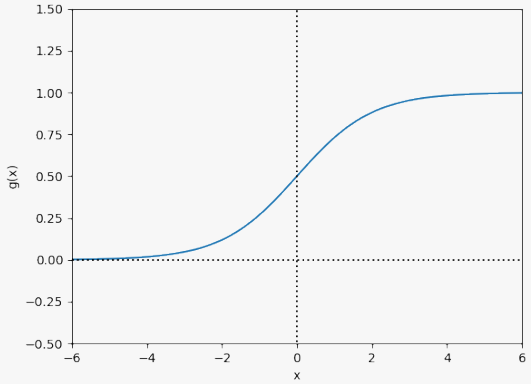


Quelques fonctions d'activation

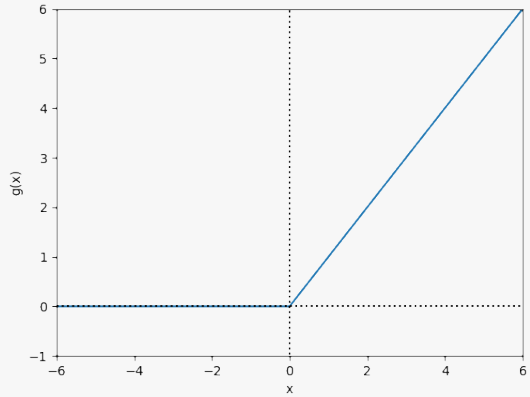
Linéaire : $g(x) = x$



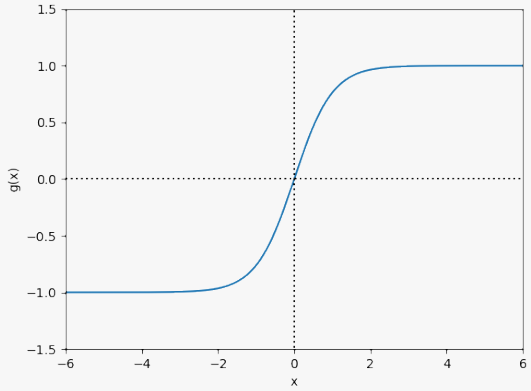
Sigmoïde : $g(x) = \frac{1}{1+\exp(-x)}$



ReLU : $g(x) = max(0, x)$



Tanh : $g(x) = \tanh(x)$

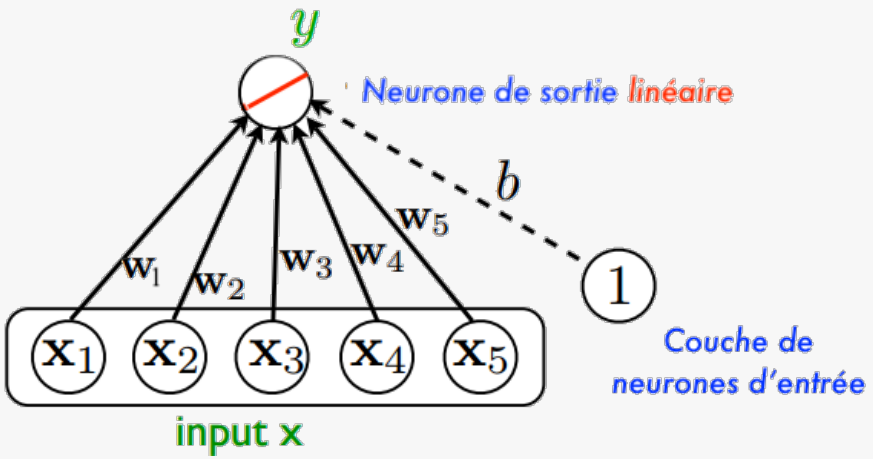


Modèle linéaire : Régression linéaire

- Modèle pour la régression

$$f(\mathrm{x}) = \mathrm{w}^T \mathrm{x} + b = \sum_i \mathrm{w}_i \mathrm{x}_i + b$$

Représentation neuronale :

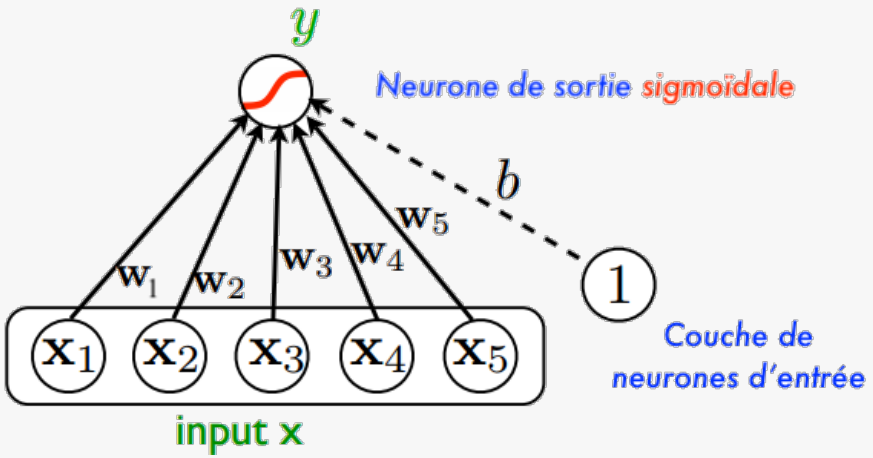


Modèle linéaire : «Régression» logistique

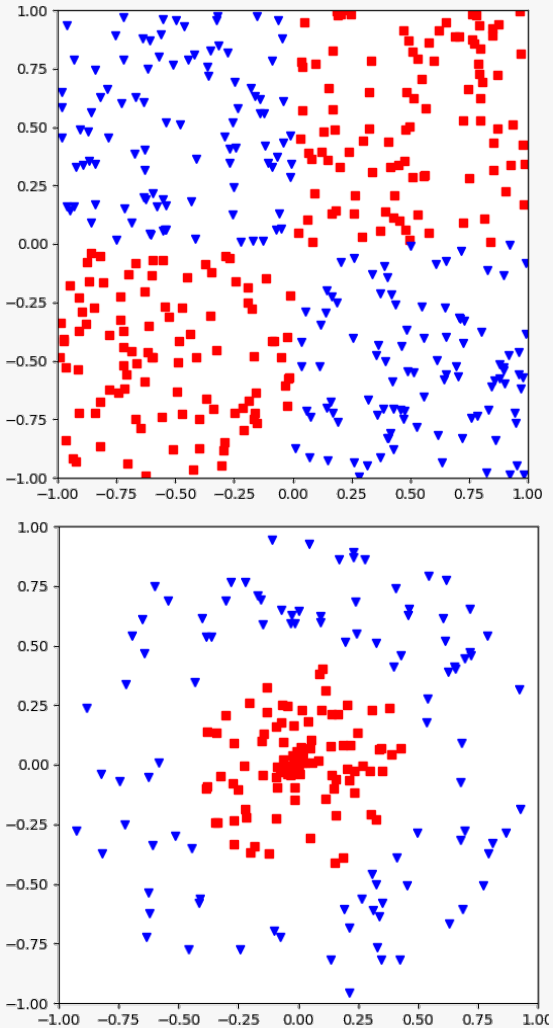
- Modèle pour la classification binaire

$$f(\mathrm{x}) = \mathrm{sigm}\left(\mathrm{w}^T\mathrm{x} + b\right) = \mathrm{sigm}\left(\sum_i \mathrm{w}_i\mathrm{x}_i + b\right)$$

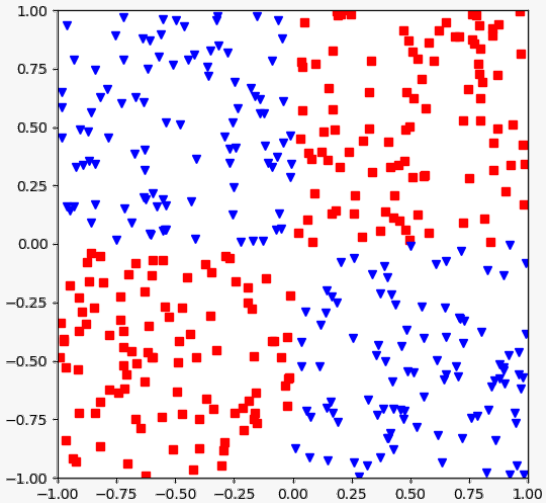
Représentation neuronale :



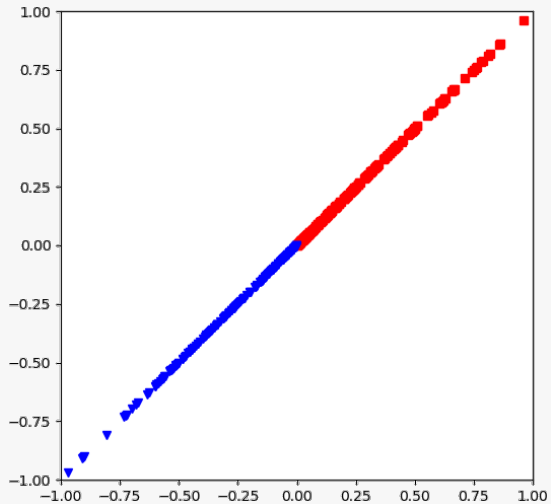
Problème : ensembles de données non linéairement séparables

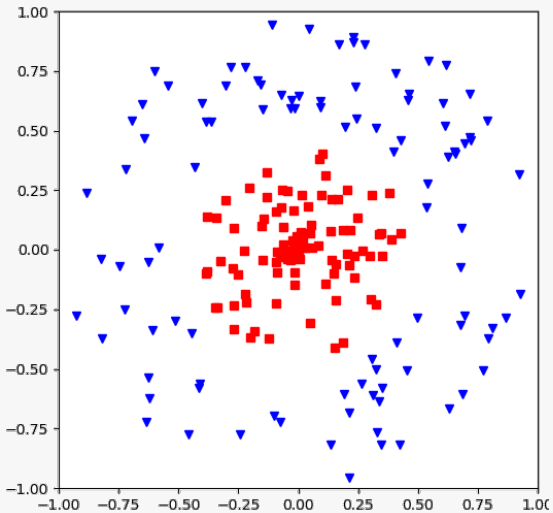


Problème : ensembles de données non linéairement séparables

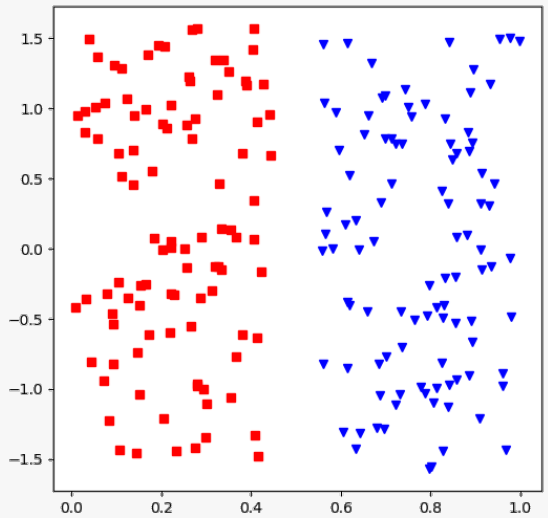


Multiplier les deux composantes →





Coordonnées polaires →



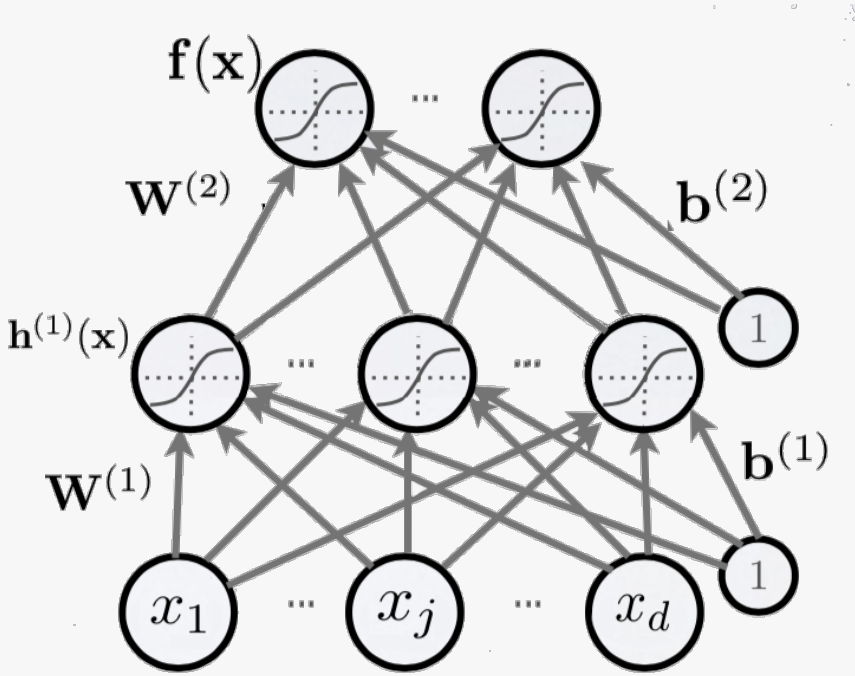
Comment définir ces nouvelles représentations pour chaque ensemble de données ?

Comment définir ces nouvelles représentations pour chaque ensemble de données ?

On laisse le modèle les apprendre !

Introduction d'une couche de neurones cachés

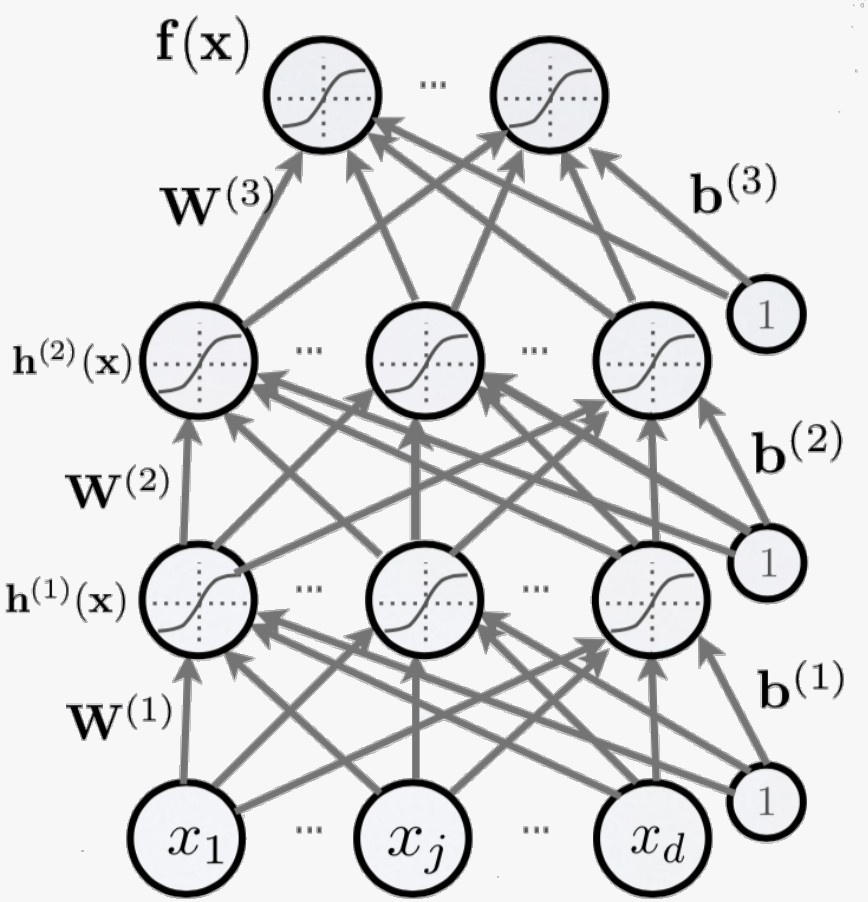
- Entrée : $\mathrm{x} \in \mathbb{R}^d$
- Paramètres :
  $W^{(1)} \in \mathbb{R}^{d \times d_h}$, $b^{(1)} \in \mathbb{R}^{d_h}$
  $W^{(2)} \in \mathbb{R}^{d_h \times d_o}$, $b^{(2)} \in \mathbb{R}^{d_o}$
- Propagation :
  $h^{(1)}(\mathrm{x}) = g^{(1)}(W^{(1),T}\mathrm{x} + b^{(1)})$
  $f(\mathrm{x}) = g^{(2)}(W^{(2),T}h^{(1)}(\mathrm{x}) + b^{(2)})$



On peut aussi avoir plus d'une couche cachée !

Ò

- C'est ce qu'on appelle l'**apprentissage profond**.
- Le nombre de couches cachées ainsi que le nombre de neurones dans chaque couche cachée sont des **hyperparamètres** que l'utilisateur doit ajuster manuellement

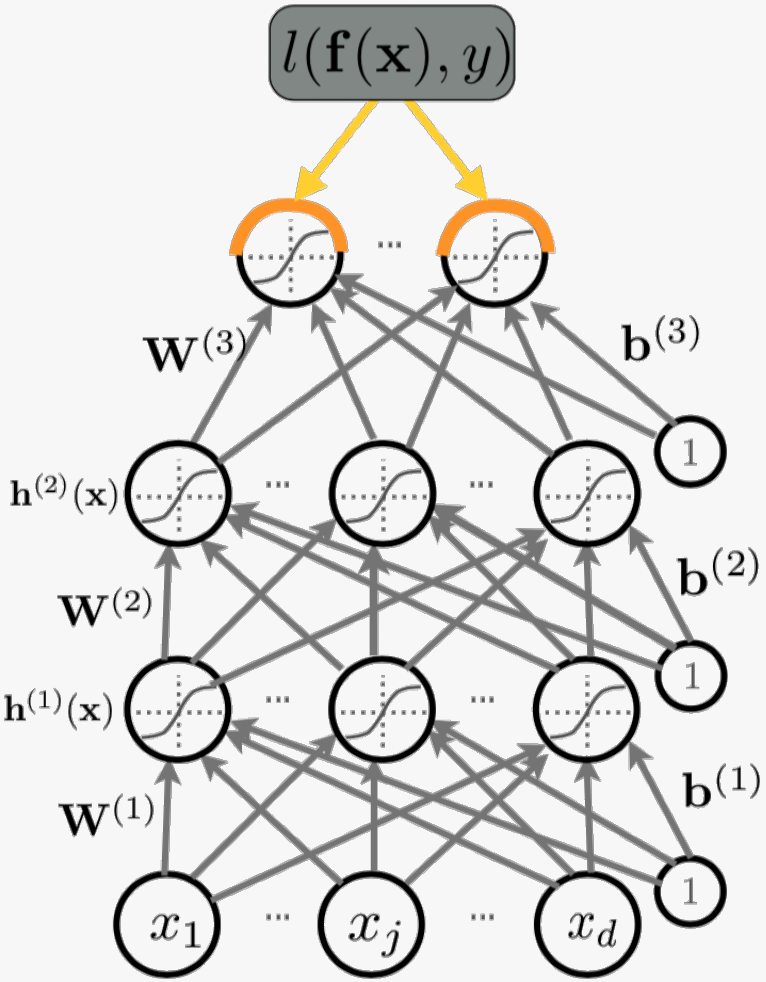


Apprentissage des paramètres : algorithme de rétropropagation[10]

- On calcule le gradient de la fonction de coût $\nabla\mathcal{L}(f(x), y)$ pour chaque élément du réseau en utilisant la règle de dérivée en chaîne.
- On ajuste les paramètres dans la direction opposée du gradient :

$$\theta^{(t+1)} \leftarrow \theta^{(t)} - \eta\nabla_\theta\mathcal{L}(f(x), y)^{(t)}$$

- Le **taux d'apprentissage** η est un hyperparamètre à déterminer.
- On répète pour chaque exemple jusqu'à convergence.

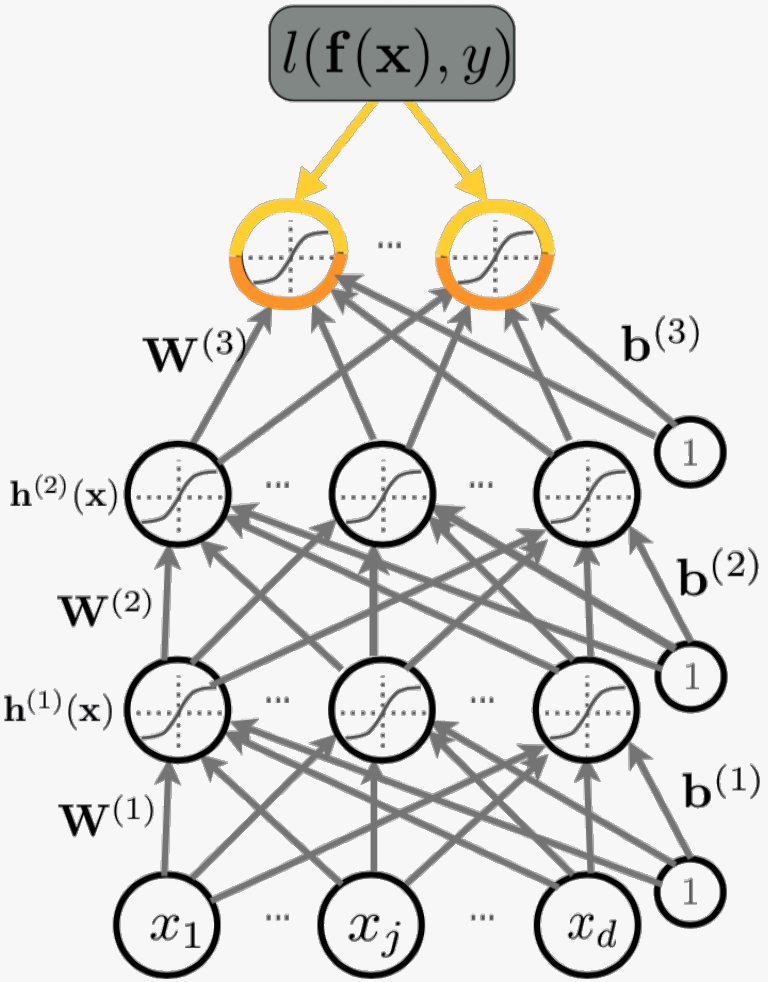


Apprentissage des paramètres : algorithme de rétropropagation[10]

- On calcule le gradient de la fonction de coût $\nabla\mathcal{L}(f(x), y)$ pour chaque élément du réseau en utilisant la règle de dérivée en chaîne.
- On ajuste les paramètres dans la direction opposée du gradient :

$$\theta^{(t+1)} \leftarrow \theta^{(t)} - \eta\nabla_\theta\mathcal{L}(f(x), y)^{(t)}$$

- Le **taux d'apprentissage** η est un hyperparamètre à déterminer.
- On répète pour chaque exemple jusqu'à convergence.

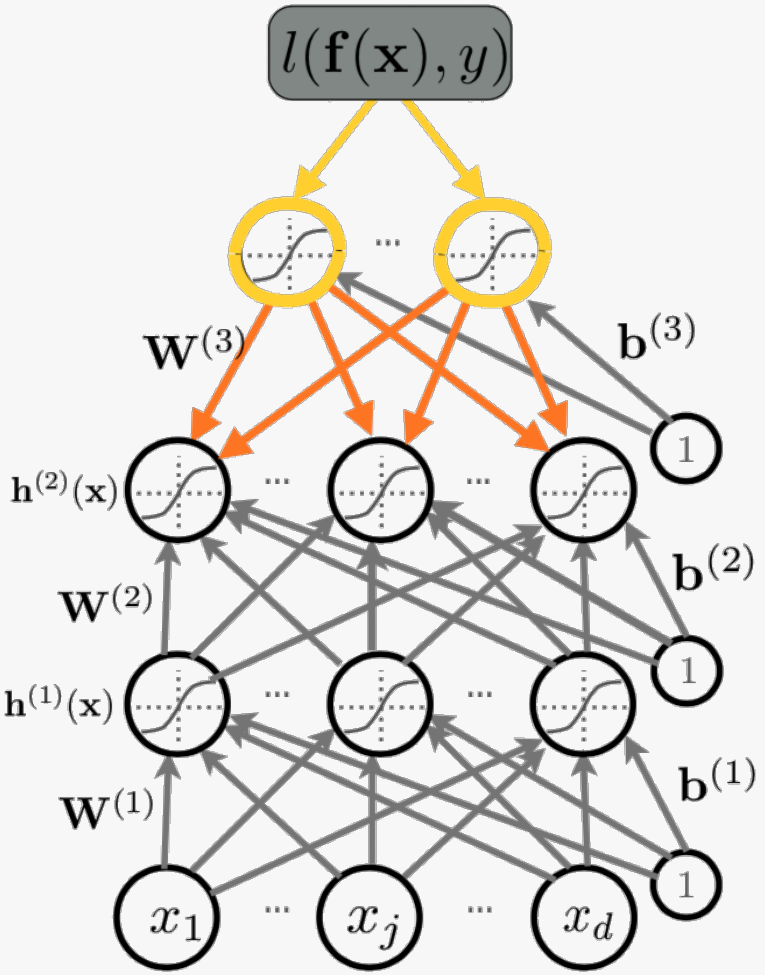


Apprentissage des paramètres : algorithme de rétropropagation[10]

- On calcule le gradient de la fonction de coût $\nabla \mathcal{L}(f(x), y)$ pour chaque élément du réseau en utilisant la règle de dérivée en chaîne.
- On ajuste les paramètres dans la direction opposée du gradient :

$$\theta^{(t+1)} \leftarrow \theta^{(t)} - \eta \nabla_\theta \mathcal{L}(f(x), y)^{(t)}$$

- Le **taux d'apprentissage** η est un hyperparamètre à déterminer.
- On répète pour chaque exemple jusqu'à convergence.

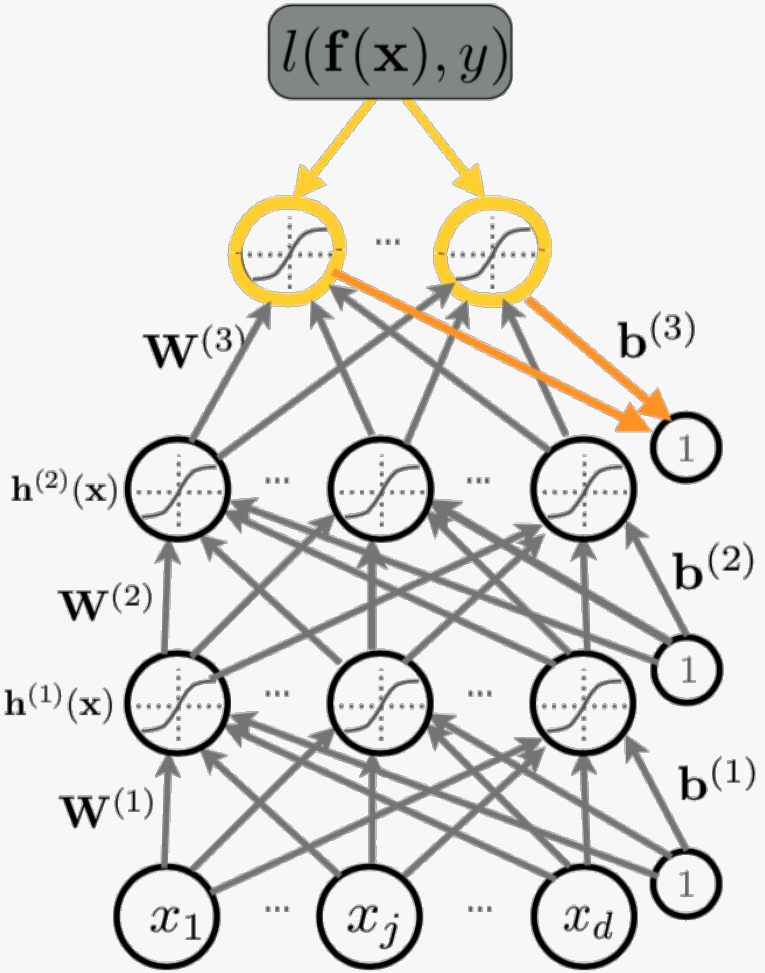


Apprentissage des paramètres : algorithme de rétropropagation[10]

- On calcule le gradient de la fonction de coût $\nabla\mathcal{L}(f(x), y)$ pour chaque élément du réseau en utilisant la règle de dérivée en chaîne.
- On ajuste les paramètres dans la direction opposée du gradient :

$$\theta^{(t+1)} \leftarrow \theta^{(t)} - \eta\nabla_\theta\mathcal{L}(f(x), y)^{(t)}$$

- Le **taux d'apprentissage** η est un hyperparamètre à déterminer.
- On répète pour chaque exemple jusqu'à convergence.

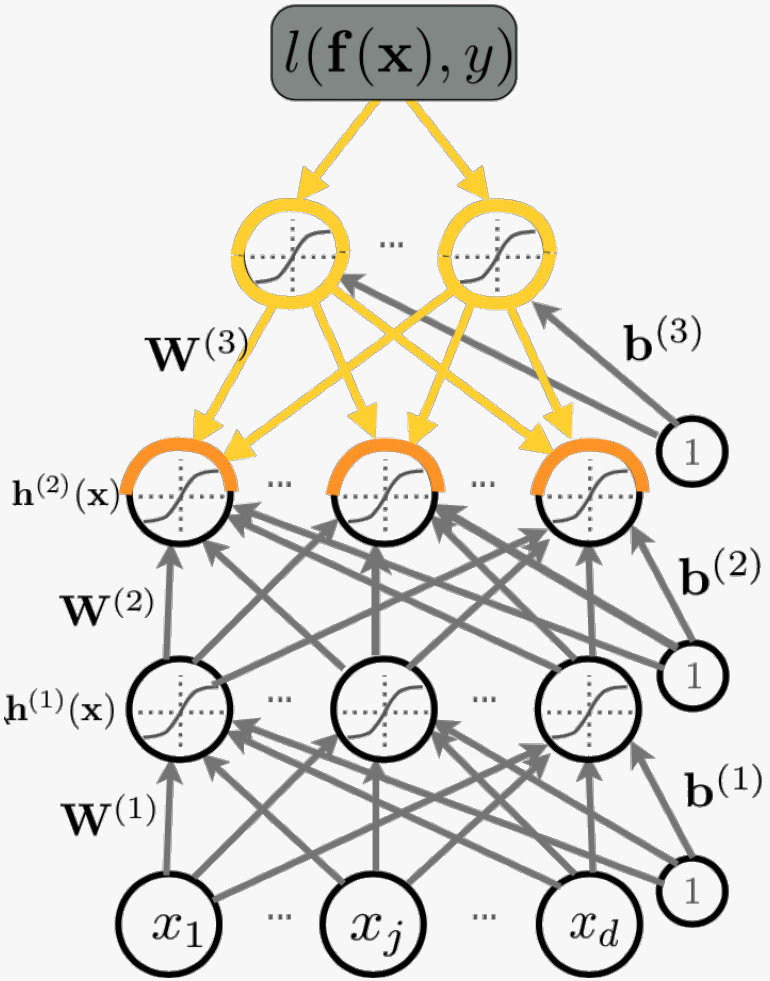


Apprentissage des paramètres : algorithme de rétropropagation[10]

- On calcule le gradient de la fonction de coût $\nabla \mathcal{L}(f(x), y)$ pour chaque élément du réseau en utilisant la règle de dérivée en chaîne.
- On ajuste les paramètres dans la direction opposée du gradient :

$$\theta^{(t+1)} \leftarrow \theta^{(t)} - \eta \nabla_\theta \mathcal{L}(f(x), y)^{(t)}$$

- Le **taux d'apprentissage** η est un hyperparamètre à déterminer.
- On répète pour chaque exemple jusqu'à convergence.

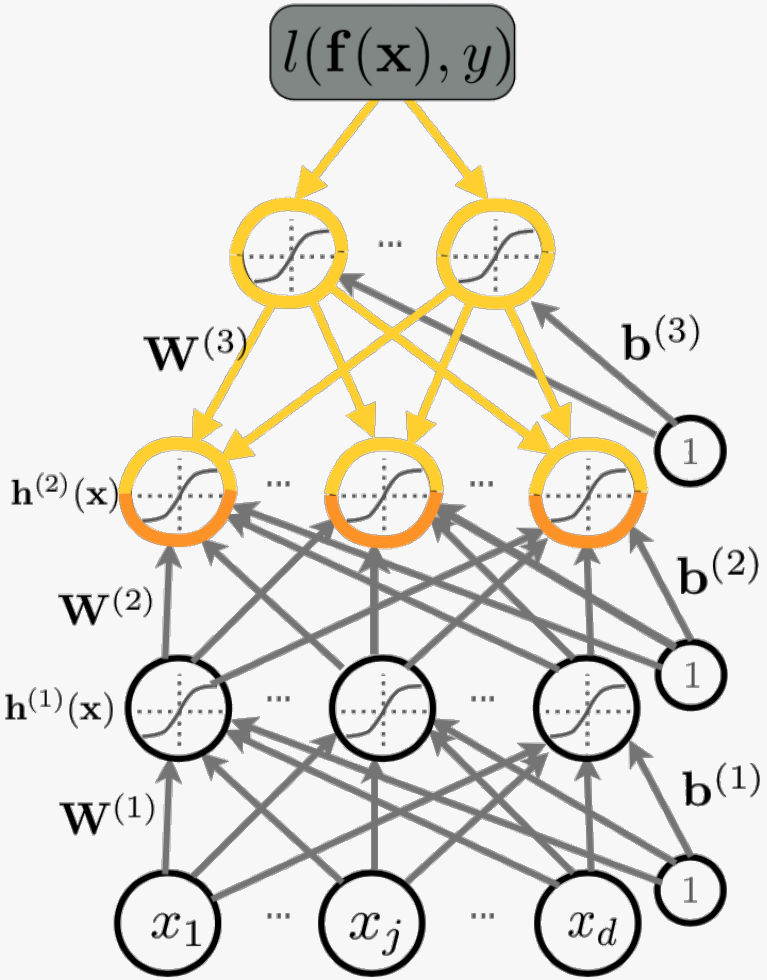


Apprentissage des paramètres : algorithme de rétropropagation[10]

- On calcule le gradient de la fonction de coût $\nabla \mathcal{L}(f(x), y)$ pour chaque élément du réseau en utilisant la règle de dérivée en chaîne.
- On ajuste les paramètres dans la direction opposée du gradient :

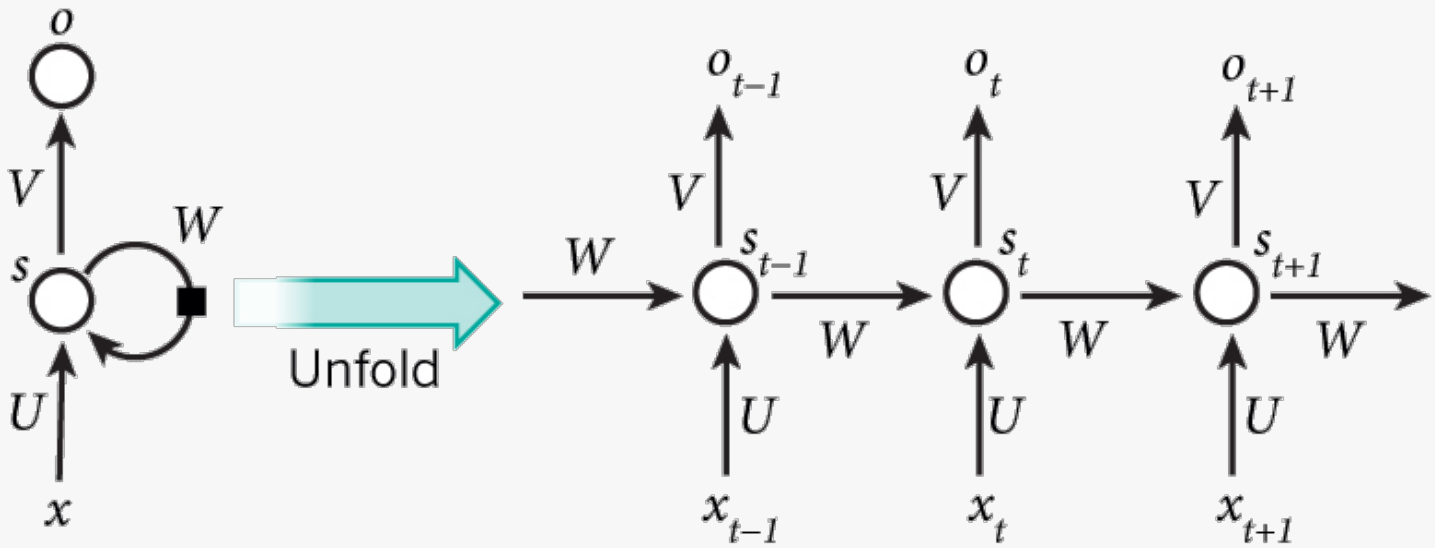
$$\theta^{(t+1)} \leftarrow \theta^{(t)} - \eta \nabla_\theta \mathcal{L}(f(x), y)^{(t)}$$

- Le **taux d'apprentissage** η est un hyperparamètre à déterminer.
- On répète pour chaque exemple jusqu'à convergence.



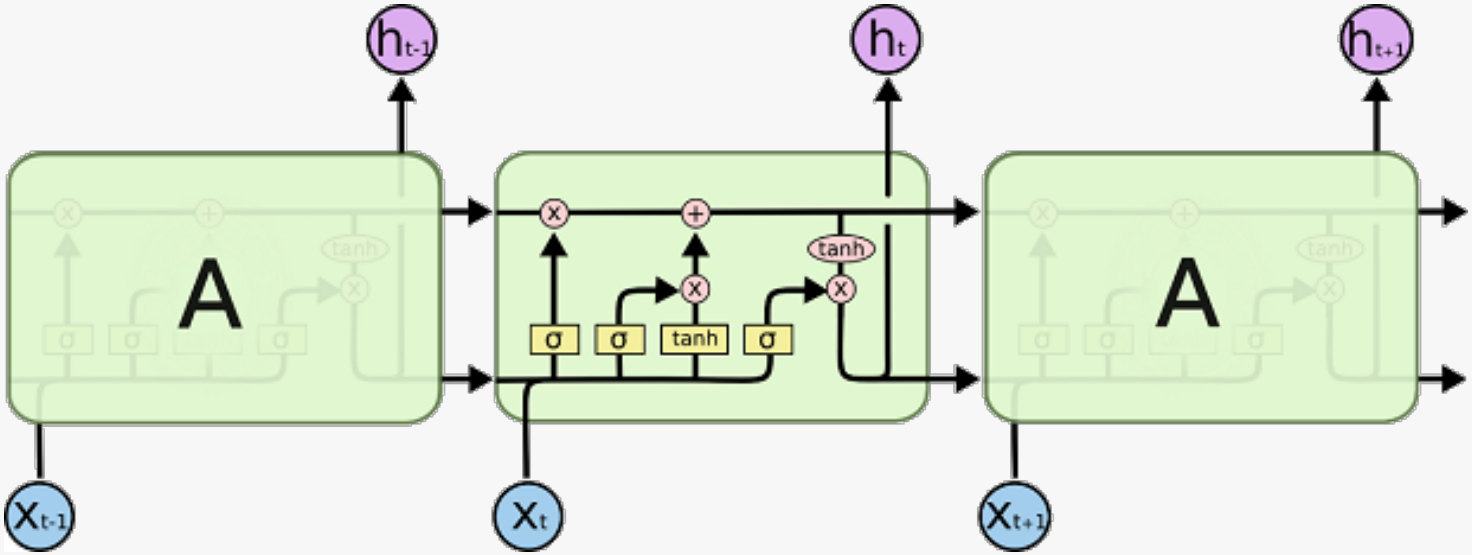
Réseaux de neurones récurrents (RNN)

- Utilisé dans les données avec corrélation temporelle et dans le traitement des langues naturelles
- Capable de manipuler des entrées de différentes longueurs
- Limitations : Difficulté avec les dépendances à long-terme



[6] pour une excellente introduction aux RNN

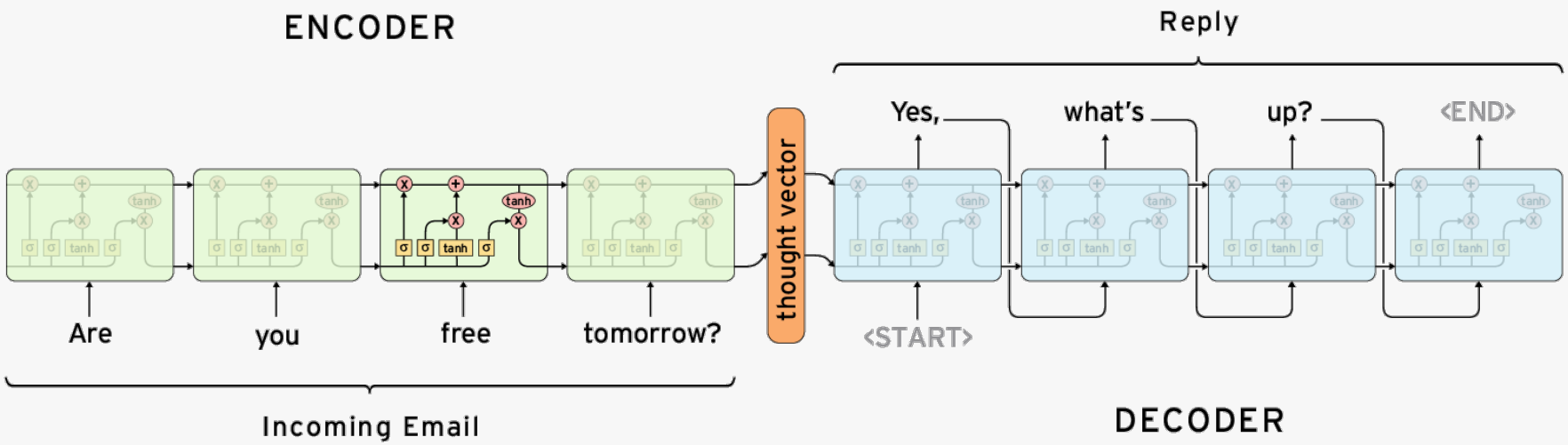
- Solution : *Long Short-Term Memory* (LSTM) [5]
- Permet au signal de se propager plus facilement dans le temps.



[9] pour une excellente introduction aux LSTM

Modèles séquence à séquence (Seq2Seq)[3][11]

- Permet de transformer une séquence de longueur arbitraire en autre séquence de longueur arbitraire



Bien représenter les mots

Bien représenter les mots

- Façon la plus simple : Vecteur *onehot*
- Vecteur de dimension V où tous les éléments sont 0 sauf un élément qui est 1.
- V : Taille du vocabulaire

Exemple :

le chat est mignon .

- **chat** $\rightarrow [0, 1, 0, ..., 0, 0]$

Approche **Sac de mots** (Bag of Words)

Pour représenter un texte, on peut simplement additionner les vecteurs :

- **le chat est mignon .** $\rightarrow [0, 1, 1, 0, ..., 0, 1, 0, 1, 0, ...0, 1]$

Sac de mots

Problèmes avec cette approche :

- Beaucoup d'espace mémoire perdue. (La majorité des composantes du vecteur sont 0)
- On pert l'ordre des mots avec cette représentation.

  "le chat mange la pizza ." $=$ **"la pizza mange le chat ."**

- Tous les mots sont indépendants les uns des autres.

  $$\textbf{chat} \perp \textbf{chien}$$

Solution possible : **Word Embeddings** [2][8]

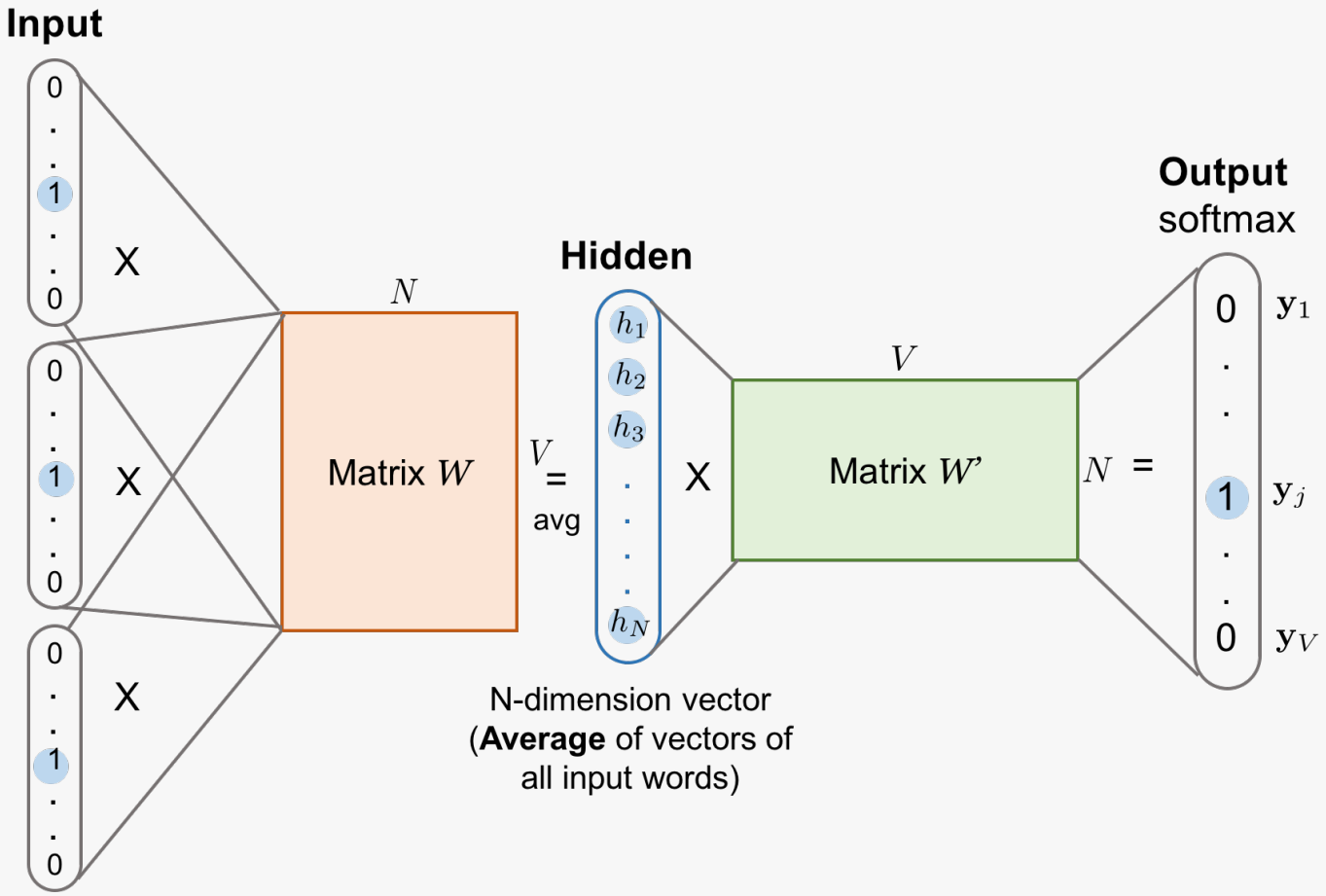
On projette les vecteurs onehot dans un espace de dimension réduite. On **apprend** cette représentation

2 approches dominantes :

- *Continuous Bag of Words (CBOW)*
- *Skip-Gram*

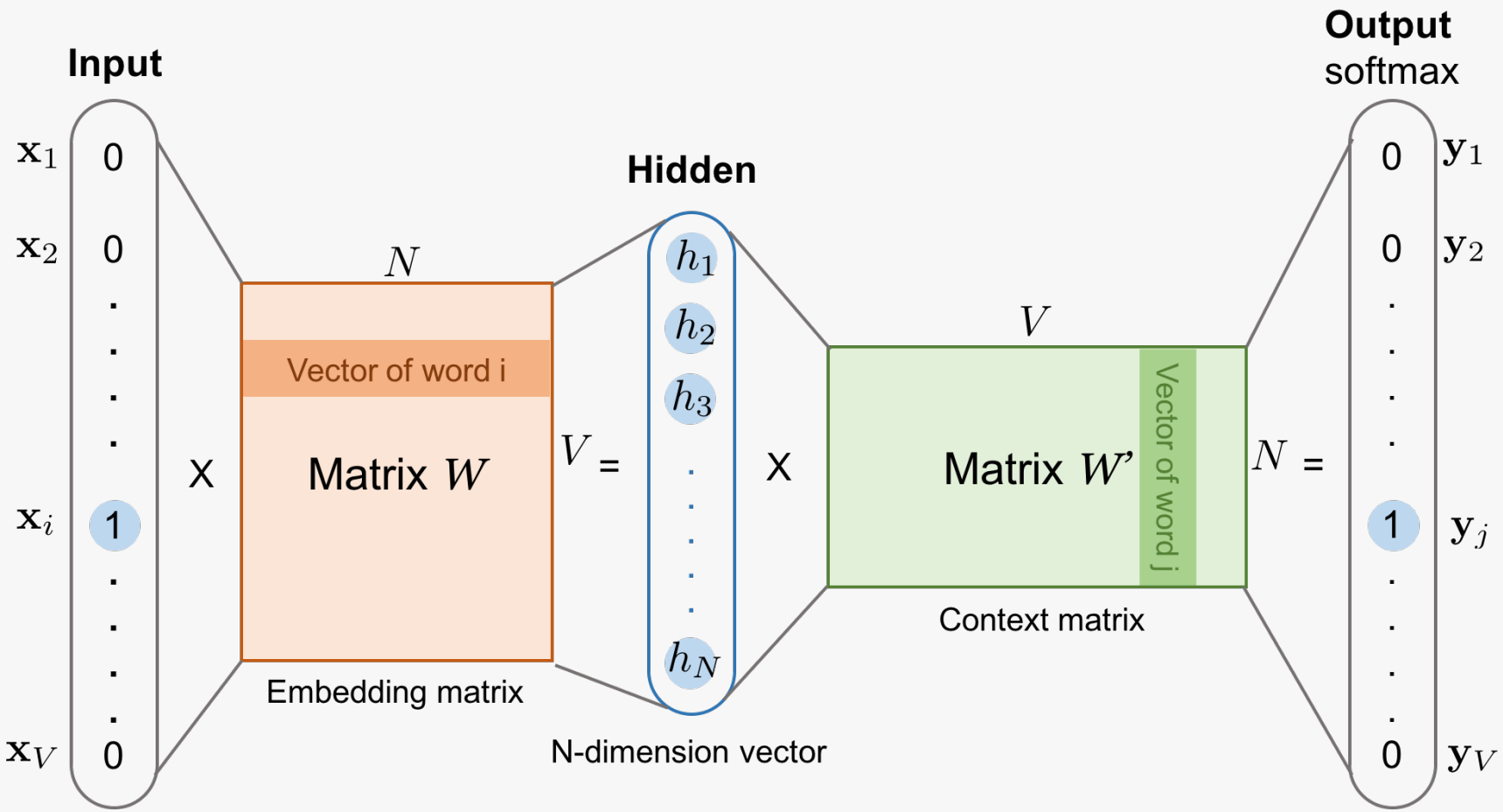
Continuous Bag of Words (CBOW)

- On utilise une fenêtre de contexte du mot et on essaie de prédire celui-ci.

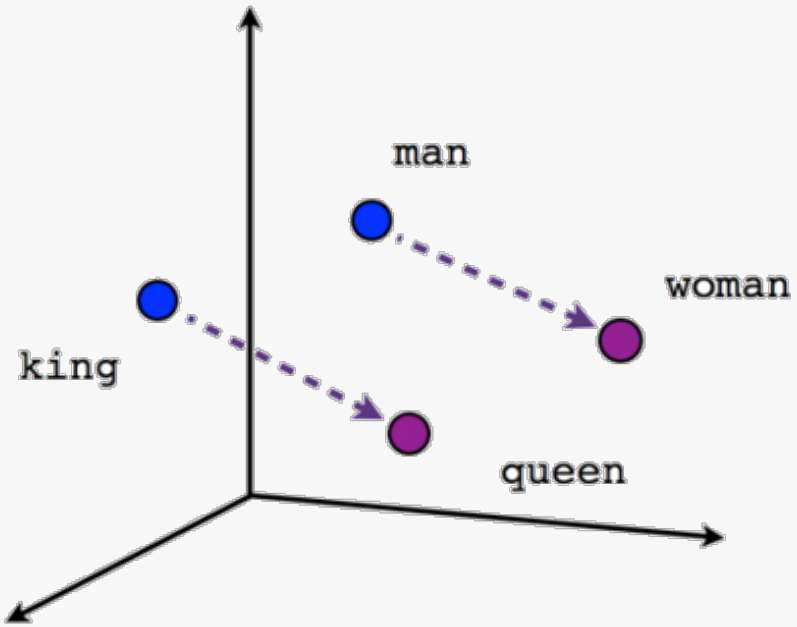


Skip-Gram

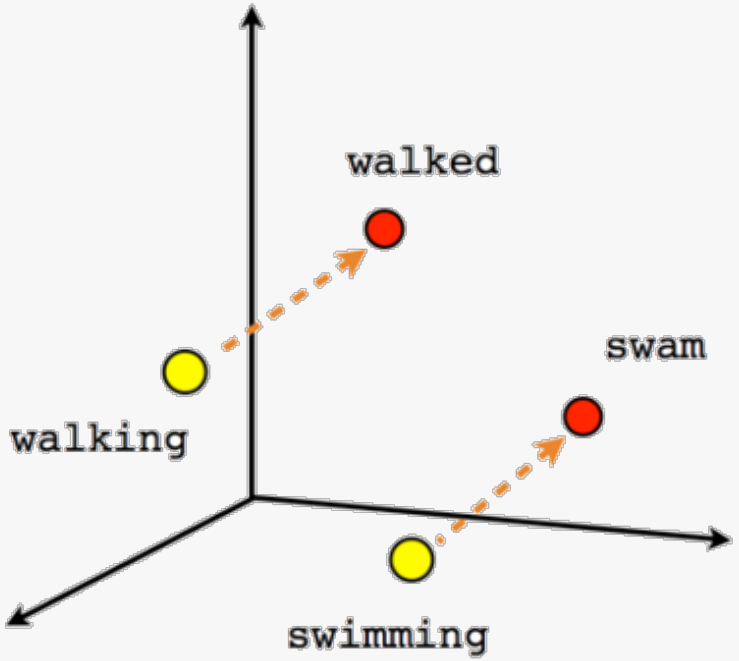
- On utilise le mot et on essaie de prédire son contexte.



$$\text{Repr}(\text{man}) - \text{Repr}(\text{woman}) + \text{Repr}(\text{King}) \approx \text{Repr}(\text{Queen})$$

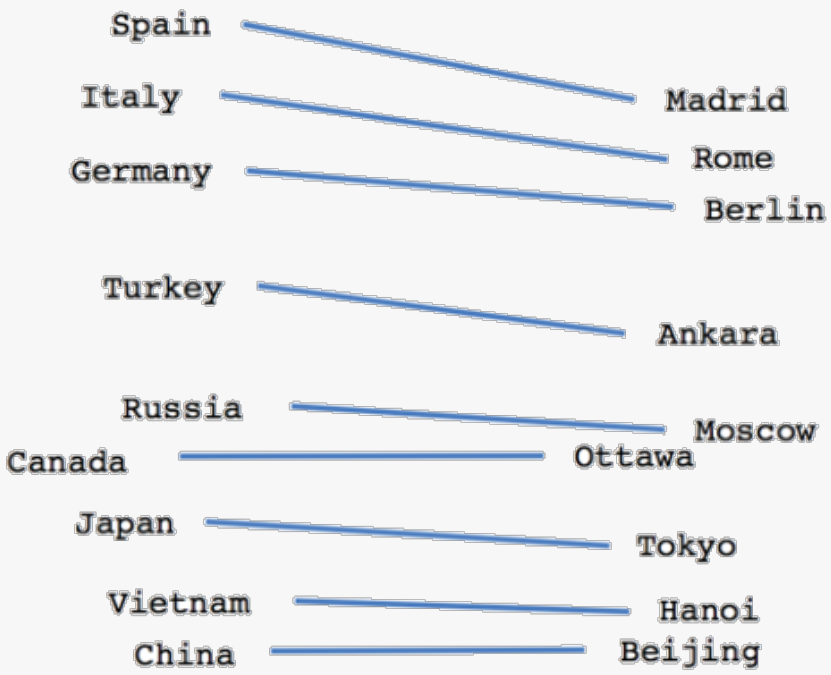


Male-Female



Verb tense

Word Embeddings



Country-Capital

Quelques tâches en NLP

Analyse de sentiments

Tâche Prédire le sentiment véhiculé dans un texte.

Exemple Prédire si une critique de film est positive ou négative.

IMDB data in the Pang and Lee database

✓

when _star wars_ came out some twenty years ago , the image of traveling throughout the stars has become a commonplace image . [...]

when han solo goes light speed , the stars change to bright lines , going towards the viewer in lines that converge at an invisible point .

cool .

october sky offers a much simpler image–that of a single white dot , traveling horizontally across the night sky . [. . .]

✗

" snake eyes " is the most aggravating kind of movie : the kind that shows so much potential then becomes unbelievably disappointing .

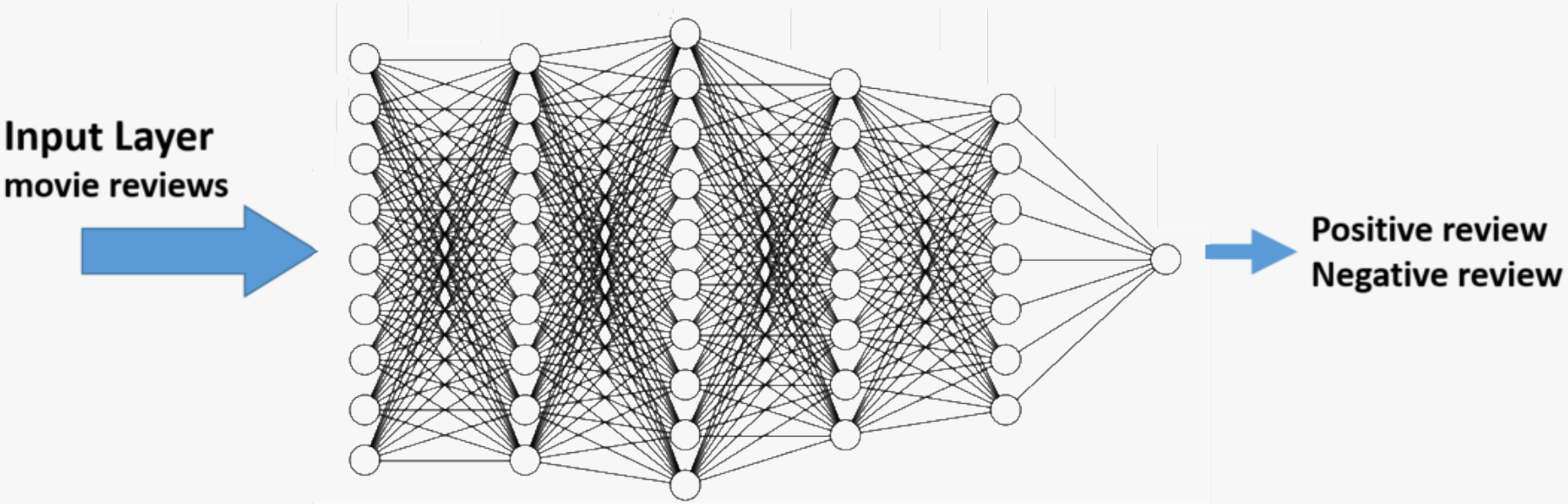
it's not just because this is a brian depalma film , and since he's a great director and one who's films are always greeted with at least some fanfare .

and it's not even because this was a film starring nicolas cage and since he gives a brauvara performance , this film is hardly worth his talents .

Analyse de sentiments

Approche simple :

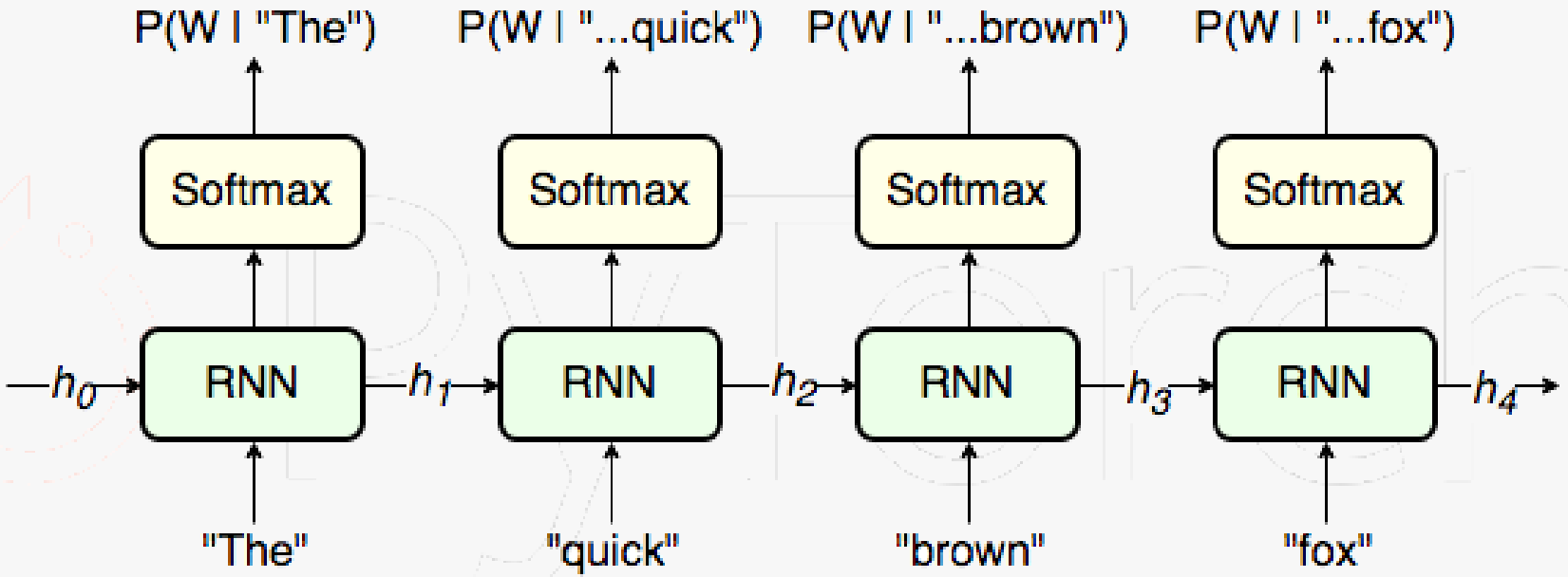
- On représente le texte d'entrée avec l'approche Bag of Words.
- On utilise un MLP pour prédire une cible binaire
- 1 : positif, 0 : négatif
- **Résultats** : environ 88% de bonnes réponses



Tâche Prédire le prochain mot en fonction des mots précédents.

Approche simple :

- On prend un mot comme cible.
- On prend comme contexte tous les mots précédant ce mot.
- On représente les mots avec des Word Embeddings.
- On utilise un RNN pour tenir compte de l'ordre temporelle des mots.



Exemple Modèle de langue imitant **Shakespeare** [6]

PANDARUS :
Alas, I think he shall be come approached and the day
When little srain would be attain'd into being never fed,
And who is but a chain and subjects of his death,
I should not sleep.

Second Senator :
They are away this miseries, produced upon my soul,
Breaking and strongly should be buried, when I perish
The earth and thoughts of many states.

DUKE VINCENTIO :
Well, your wit is in the care of side and that.

Exemple Modèle de langue imitant des articles de géométrie algébrique en LaTeX[6]

For $\bigoplus_{n=1,\ldots,m}$ where $\mathcal{L}_{m_\bullet} = 0$, hence we can find a closed subset $\mathcal{H}$ in $\mathcal{H}$ and any sets $\mathcal{F}$ on X, U is a closed immersion of S, then $U \to T$ is a separated algebraic space.

Proof. Proof of (1). It also start we get

$$S = \mathrm{Spec}(R) = U \times_X U \times_X U$$

and the compariciely in the fibre product covering we have to prove the lemma generated by $\coprod Z \times_U U \to V$. Consider the maps M along the set of points Sch_{fppf} and $U \to U$ is the fibre category of S in U in Section, ?? and the fact that any U affine, see Morphisms, Lemma ??. Hence we obtain a scheme S and any open subset $W \subset U$ in $Sh(G)$ such that $\mathrm{Spec}(R') \to S$ is smooth or an

$$U = \bigcup U_i \times_{S_i} U_i$$

which has a nonzero morphism we may assume that f_i is of finite presentation over S. We claim that $\mathcal{O}_{X,x}$ is a scheme where $x, x', s'' \in S'$ such that $\mathcal{O}_{X,x'} \to \mathcal{O}'_{X',x'}$ is separated. By Algebra, Lemma ?? we can define a map of complexes $\mathrm{GL}_{S'}(x'/S'')$ and we win. □

To prove study we see that $\mathcal{F}|_U$ is a covering of $\mathcal{X}'$, and $\mathcal{T}_i$ is an object of $\mathcal{F}_{X/S}$ for $i > 0$ and $\mathcal{F}_p$ exists and let $\mathcal{F}_i$ be a presheaf of $\mathcal{O}_X$-modules on $\mathcal{C}$ as a $\mathcal{F}$-module. In particular $\mathcal{F} = U/\mathcal{F}$ we have to show that

$$\widetilde{M^\bullet} = \mathcal{I}^\bullet \otimes_{\mathrm{Spec}(k)} \mathcal{O}_{S,s} - i_X^{-1}\mathcal{F})$$

is a unique morphism of algebraic stacks. Note that

$$\mathrm{Arrows} = (Sch/S)^{opp}_{fppf}, (Sch/S)_{fppf}$$

and

$$V = \Gamma(S, \mathcal{O}) \longmapsto (U, \mathrm{Spec}(A))$$

is an open subset of X. Thus U is affine. This is a continuous map of X is the inverse, the groupoid scheme S.

Proof. See discussion of sheaves of sets. □

The result for prove any open covering follows from the less of Example ??. It may replace S by $X_{spaces,étale}$ which gives an open subspace of X and T equal to S_{Zar}, see Descent, Lemma ??. Namely, by Lemma ?? we see that R is geometrically regular over S.

Lemma 0.1. *Assume (3) and (3) by the construction in the description.*

Suppose $X = \lim |X|$ *(by the formal open covering* X *and a single map* $\underline{Proj}_X(\mathcal{A}) = \mathrm{Spec}(B)$ *over* U *compatible with the complex*

$$Set(\mathcal{A}) = \Gamma(X, \mathcal{O}_{X,\mathcal{O}_X}).$$

When in this case of to show that $\mathcal{Q} \to \mathcal{C}_{Z/X}$ *is stable under the following result in the second conditions of (1), and (3). This finishes the proof. By Definition ?? (without element is when the closed subschemes are catenary. If* T *is surjective we may assume that* T *is connected with residue fields of* S*. Moreover there exists a closed subspace* $Z \subset X$ *of* X *where* U *in* X' *is proper (some defining as a closed subset of the uniqueness it suffices to check the fact that the following theorem*

(1) f *is locally of finite type. Since* $S = \mathrm{Spec}(R)$ *and* $Y = \mathrm{Spec}(R)$.

Proof. This is form all sheaves of sheaves on X. But given a scheme U and a surjective étale morphism $U \to X$. Let $U \cap U = \coprod_{i=1,\ldots,n} U_i$ be the scheme X over S at the schemes $X_i \to X$ and $U = \lim_i X_i$. □

The following lemma surjective restrocomposes of this implies that $\mathcal{F}_{x_0} = \mathcal{F}_{x_0} = \mathcal{F}_{\mathcal{X},\ldots,0}$.

Lemma 0.2. *Let* X *be a locally Noetherian scheme over* S, $E = \mathcal{F}_{X/S}$. *Set* $\mathcal{I} = \mathcal{J}_1 \subset \mathcal{I}'_n$. *Since* $\mathcal{I}^n \subset \mathcal{I}^n$ *are nonzero over* $i_0 \leq \mathfrak{p}$ *is a subset of* $\mathcal{J}_{n,0} \circ \overline{A}_2$ *works.*

Lemma 0.3. *In Situation ??. Hence we may assume* $\mathfrak{q}' = 0$.

Proof. We will use the property we see that $\mathfrak{p}$ is the mext functor (??). On the other hand, by Lemma ?? we see that

$$D(\mathcal{O}_{X'}) = \mathcal{O}_X(D)$$

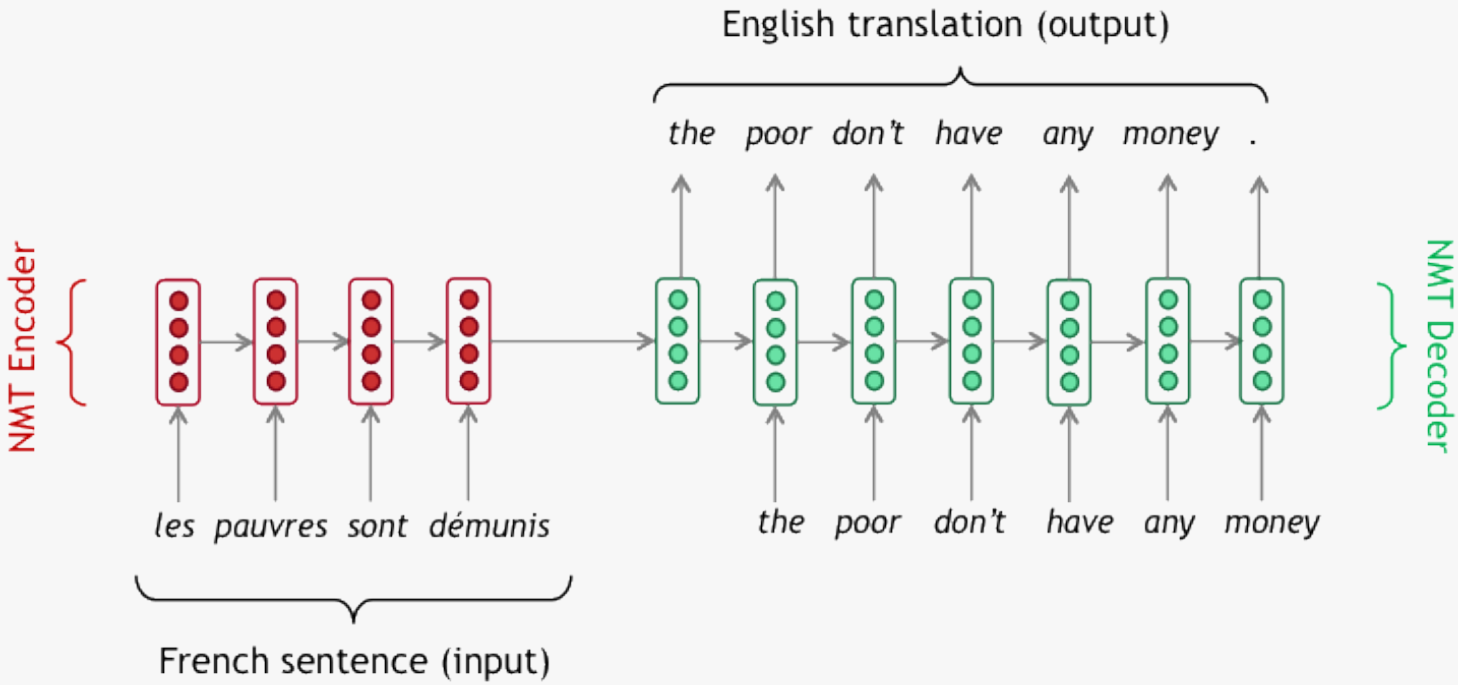
where K is an F-algebra where δ_{n+1} is a scheme over S. □

Traduction automatique

Tâche Traduire une phrase d'une langue source vers une langue cible.

Approche simple :

- On utilise un modèle Seq2Seq.
- On utilise des word embeddings pour représenter les mots.



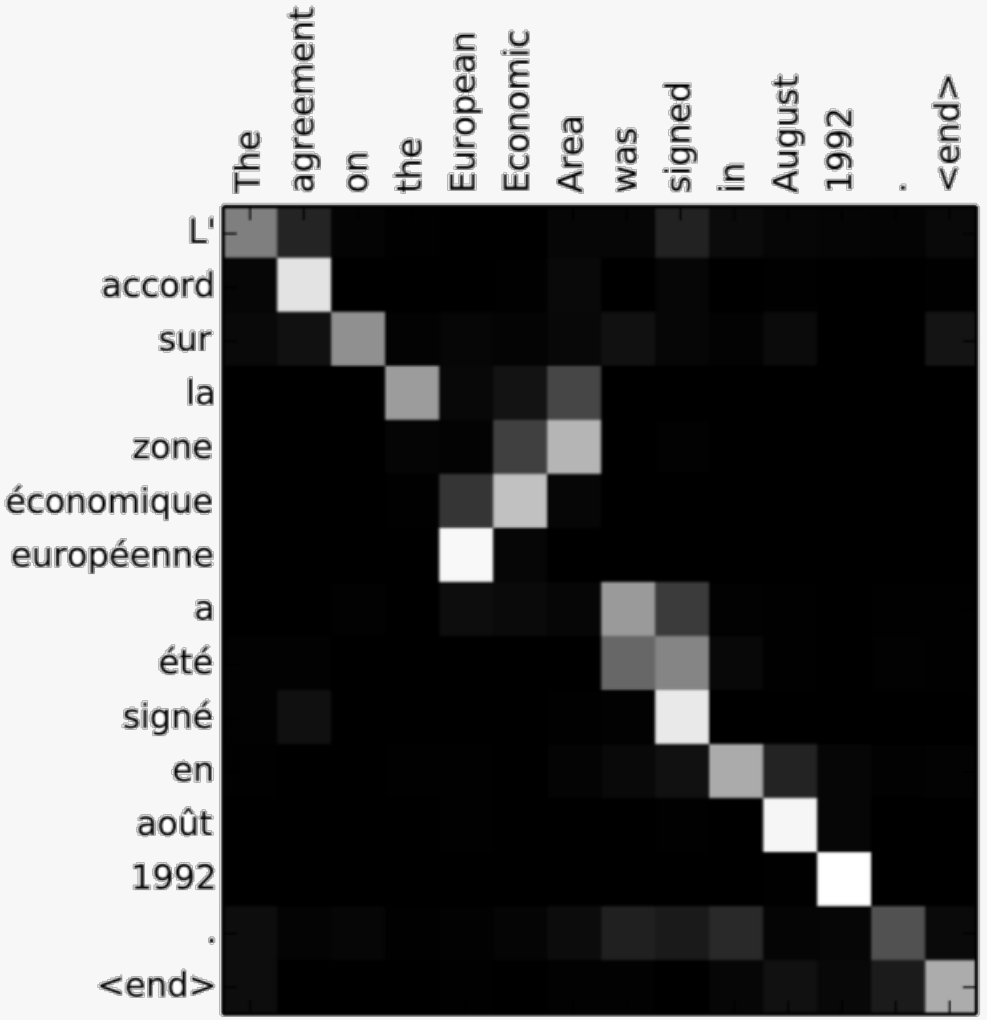
Exemple Traduction anglais vers français [11]

- **Traduction générée** : Avec la crémation , il y a un " sentiment de violence contre le corps d ' un être cher " , qui sera " réduit à une pile de cendres " en très peu de temps au lieu d ' un processus de décomposition " qui accompagnera les étapes du deuil " .
- **Cible** : Il y a , avec la crémation , " une violence faite au corps aimé " , qui va être " réduit à un tas de cendres " en très peu de temps , et non après un processus de décomposition , qui " accompagnerait les phases de deuil " .

Limitations Toute l'information contenue dans la phrase source est transmise au décodeur par un seul vecteur de taille fixe. Les modèles ont alors tendance à avoir de la difficulté à traduire de longues phrases.

Solution : **Mécanisme d'attention** [1]

- Le modèle apprend à aligner les mots de la phrase d'entrée avec ceux de la phrase de sortie.

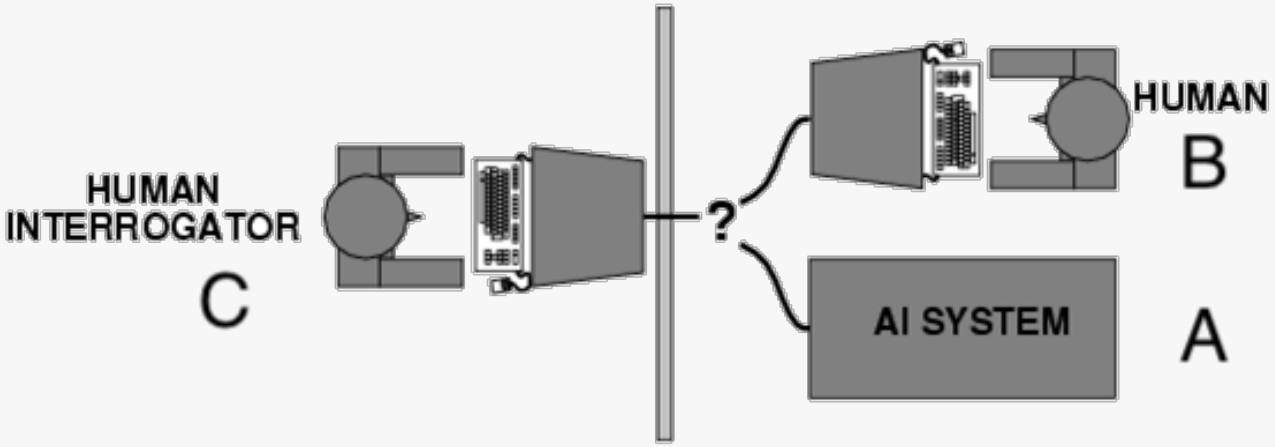


Agents conversationnels

- Communément appelés *chatbots*
- Tâche similaire à la traduction automatique.

Test de Turing [12] :

- Le but de A est de tromper l'interrogateur (C).
- Le but de B est d'aider C.
- Si une machine remplace A, puis C prédit à 50% qui est qui, le test est passé.



Exemple Réponses générées par un modèle Seq2Seq simple [13]

Human	hello !
Machine	hello !
Human	how are you ?
Machine	i 'm good .
Human	what 's your name ?
Machine	i 'm julia .
Human	when were you born ?
Machine	1977 .
Human	where are you from ?
Machine	i 'm out in the boonies.
Human	are you a man or a woman ?
Machine	i 'm a woman .
Human	why are we here ?
Machine	i 'm not sure .
Human	okay, bye !
Machine	bye .

Difficultés :

- Les réponses générées sont courtes et peu originales.
- L'agent conversationnel n'a pas de personnalité cohérente.

Conclusion

Introduction à divers modèles neuronaux utilisés dans le NLP :

- MLP, RNN, Seq2Seq

Différentes façons de représenter les mots :

- Bag of Words, Word Embeddings

Présentation de quelques tâches en NLP :

- Analyse de sentiments, Modèles de langue, Traduction automatique, Chatbots

Pour aller plus loin...

Blog posts informatifs sur le NLP et l'apprentissage profond :

- Karpathy : The Unreasonable Effectiveness of Recurrent Neural Networks
- Olah : Understanding LSTM Networks
- Brownlee : A Gentle Introduction to the Bag-of-Words Model
- Brownlee : What Are Word Embeddings for Text?
- Robertson : Translation with a Sequence to Sequence Network and Attention

Librairies Python pour le NLP et l'apprentissage profond :

- Natural Language Tool Kit (NLTK)
- PyTorch
- OpenNMT-Py

Références I

[1] D. Bahdanau, K. Cho, and Y. Bengio, *Neural machine translation by jointly learning to align and translate*, arXiv preprint arXiv :1409.0473, (2014).

[2] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, *A neural probabilistic language model*, Journal of machine learning research, 3 (2003), pp. 1137–1155.

[3] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, *Learning phrase representations using rnn encoder-decoder for statistical machine translation*, arXiv preprint arXiv :1406.1078, (2014).

[4] D. Harrison Jr and D. L. Rubinfeld, *Hedonic housing prices and the demand for clean air*, Journal of environmental economics and management, 5 (1978), pp. 81–102.

Références II

[5] S. Hochreiter and J. Schmidhuber, *Long short-term memory*, Neural computation, 9 (1997), pp. 1735–1780.

[6] A. Karpathy, *The unreasonable effectiveness of recurrent neural networks*, Andrej Karpathy blog, (2015).

[7] Y. LeCun, C. Cortes, and C. Burges, *Mnist handwritten digit database*, AT&T Labs [Online]. Available : http ://yann. lecun. com/exdb/mnist, 2 (2010).

[8] T. Mikolov, K. Chen, G. Corrado, and J. Dean, *Efficient estimation of word representations in vector space*, arXiv preprint arXiv :1301.3781, (2013).

[9] C. Olah, *Understanding lstm networks*, GITHUB blog, posted on August, 27 (2015), p. 2015.

Références III

[10] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, *Learning internal representations by error propagation*, tech. rep., California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[11] I. Sutskever, O. Vinyals, and Q. V. Le, *Sequence to sequence learning with neural networks*, in Advances in neural information processing systems, 2014, pp. 3104–3112.

[12] A. M. Turing, *Computing machinery and intelligence*, in Parsing the Turing Test, Springer, 2009, pp. 23–65.

[13] O. Vinyals and Q. Le, *A neural conversational model*, arXiv preprint arXiv :1506.05869, (2015).