Tutoriel sur les réseaux de neurones en traitement de la langue

Hugo Larochelle

26 Avril 2006

Didactiel sur les réseaux de neurones en traitement de la langue

Hugo Larochelle

26 Avril 2006

Tutaurielle sur les réseaux de neurones en traitement de la langue

Hugo Larochelle

26 Avril 2006

Plan

1. Introduction
2. Réseaux de Neurones
3. Représentations Distribuées
4. Applications
   - Modélisation de la langue
   - Reconnaissance de la parole
   - Recherche d'Information
   - Classification syntaxique
   - Autres
5. Conclusion

Plan

1. Introduction
2. Réseaux de Neurones
3. Représentations Distribuées
4. Applications
   - Modélisation de la langue
   - Reconnaissance de la parole
   - Recherche d'Information
   - Classification syntaxique
   - Autres
5. Conclusion

Les enjeux

- Un problème récurrent aux tâches de traitement de la langue est le manque de données :
  “In language applications, there is always too little data, even when hundreds of thousands of words are available.” (Sutton and McCallum, 2006)

Les enjeux

- Un problème récurrent aux tâches de traitement de la langue est le manque de données :
  “In language applications, there is always too little data, even when hundreds of thousands of words are available.” (Sutton and McCallum, 2006)
- Ceci est la conséquence directe de la grande dimensionalité des données langagières, où on a vraiment une dimension pour chaque mot différent (e.g. codage “one-hot”)

Les enjeux

- Un problème récurrent aux tâches de traitement de la langue est le manque de données :
  “In language applications, there is always too little data, even when hundreds of thousands of words are available.” (Sutton and McCallum, 2006)
- Ceci est la conséquence directe de la grande dimensionalité des données langagières, où on a vraiment une dimension pour chaque mot différent (e.g. codage “one-hot”)
- Ainsi, l’apprentissage de représentations plus **compactes** et **informatives** est fortement justifiée

Les enjeux

- Les réseaux de neurones se prêtent très bien à cette approche

Les enjeux

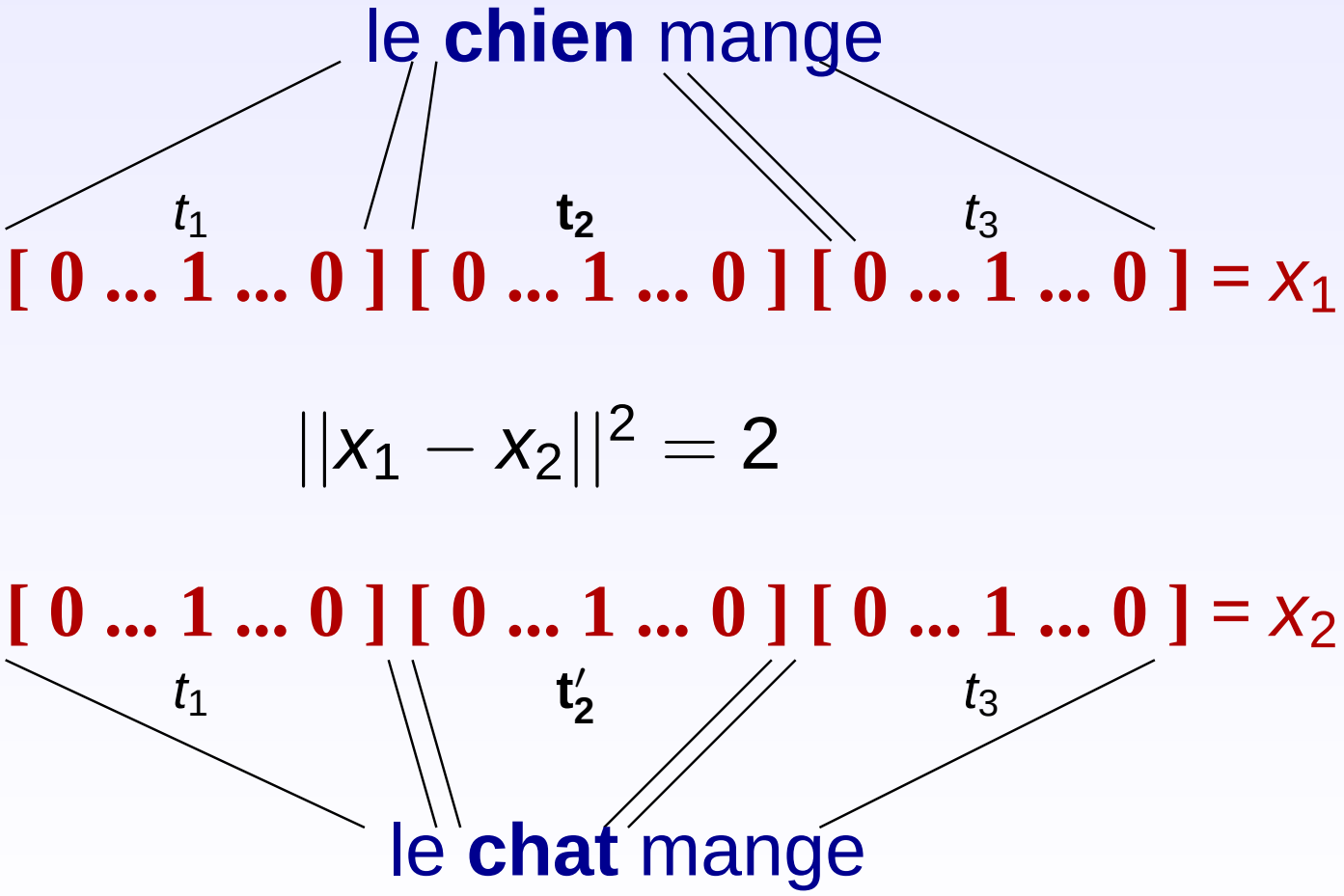
- Les réseaux de neurones se prêtent très bien à cette approche
- On va se concentrer sur les tâches (fonctions) qui peuvent être vue sous la forme suivante :

$$w_1^N \mapsto y_1^M$$

Ceci inclut la modélisation de la langue, la traduction, la classification de mots ou de documents, etc.

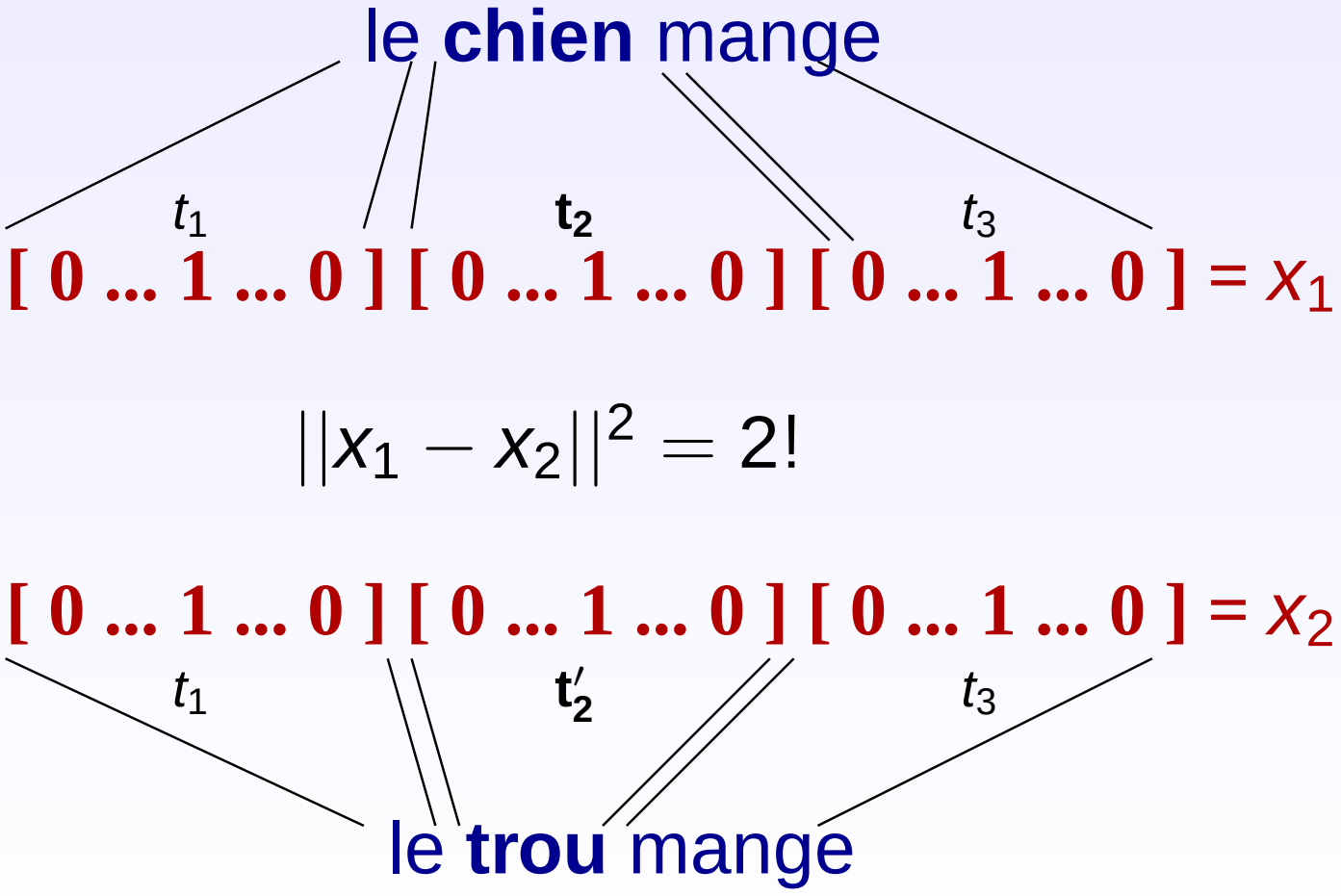
Pourquoi les représentations des mots

Distance entre deux phrases, avec codage “one-hot” :



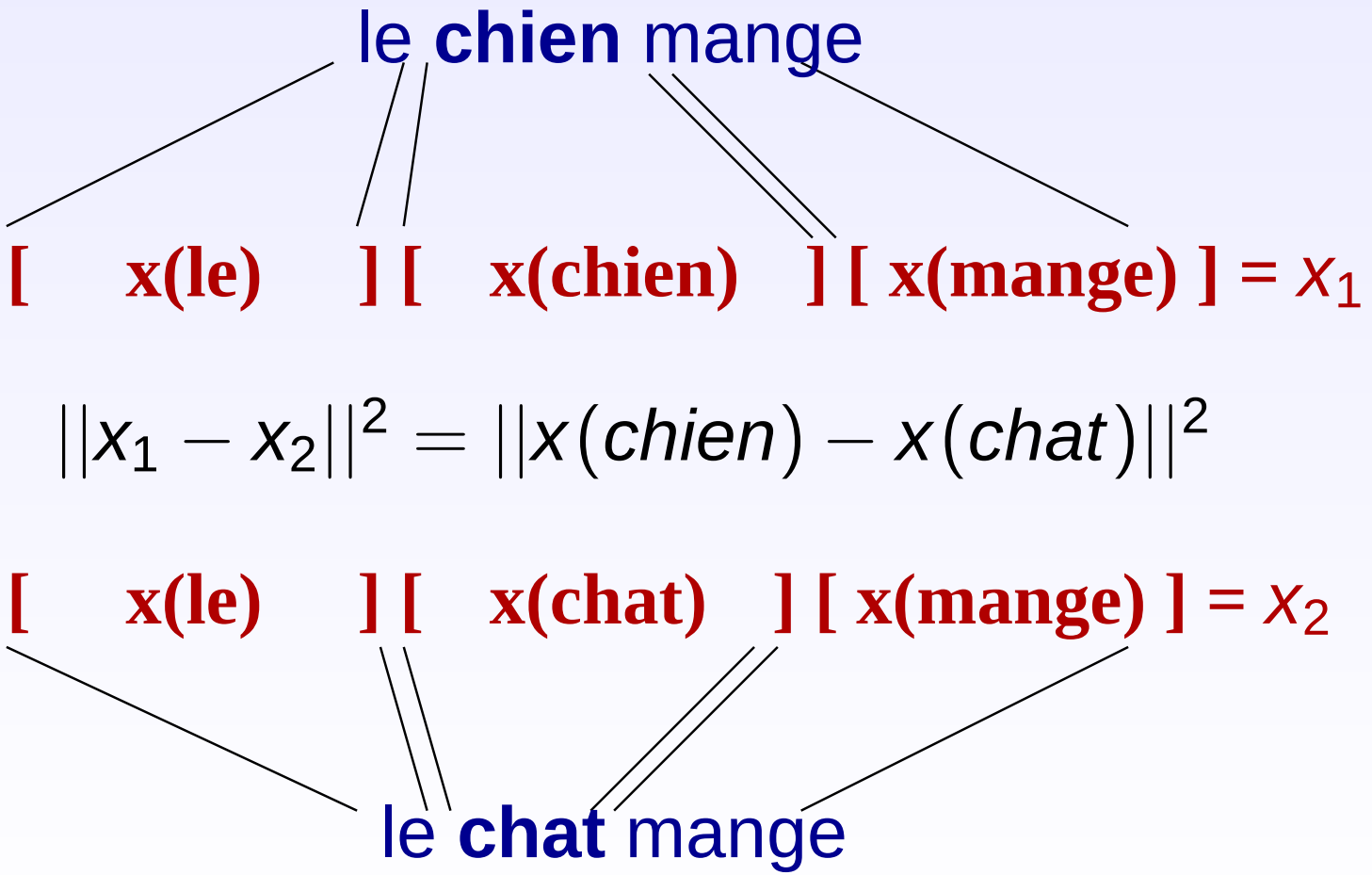
Pourquoi les représentations des mots

Même distance sous codage “one-hot” !



Pourquoi les représentations des mots

À l'aide de représentations $x(\cdot)$, on peut palier ce problème :



Plan

1. Introduction
2. Réseaux de Neurones
3. Représentations Distribuées
4. Applications
   - Modélisation de la langue
   - Reconnaissance de la parole
   - Recherche d'Information
   - Classification syntaxique
   - Autres
5. Conclusion

Défi nition mathématique

- Un réseau de neurones (à une couche cachée) est simplement une fonction ayant la forme suivante :

$$f(x) = h(\, o(x)\,) \qquad \text{avec} \qquad o_i(x) = \sum_j W_{ij} a_j(x)$$

$$a_j(x) = g(\sum_k V_{jk} x_k)$$

où $g(\cdot)$ et $h(\cdot)$ sont des fonctions d'activation (sigmoid, tanh, softmax)

Défi nition mathématique

- Un réseau de neurones (à une couche cachée) est simplement une fonction ayant la forme suivante :

$$f(x) = h(\, o(x)\,) \qquad \text{avec} \qquad o_i(x) = \sum_j W_{ij} a_j(x)$$

$$a_j(x) = g(\sum_k V_{jk} x_k)$$

  où $g(\cdot)$ et $h(\cdot)$ sont des fonctions d'activation (sigmoid, tanh, softmax)
- Les réseaux de neurones peuvent approximer la plupart des fonctions, donc peuvent résoudre la plupart des tâches

Défi nition mathématique

- Un réseau de neurones (à une couche cachée) est simplement une fonction ayant la forme suivante :

$$f(x) = h(\, o(x)\,) \qquad \text{avec} \qquad o_i(x) = \sum_j W_{ij} a_j(x)$$

$$a_j(x) = g(\sum_k V_{jk} x_k)$$

  où $g(\cdot)$ et $h(\cdot)$ sont des fonctions d'activation (sigmoid, tanh, softmax)
- Les réseaux de neurones peuvent approximer la plupart des fonctions, donc peuvent résoudre la plupart des tâches
- Tout ce qui manque, c'est un moyen d'obtenir les matrices W et V

Défi nition mathématique

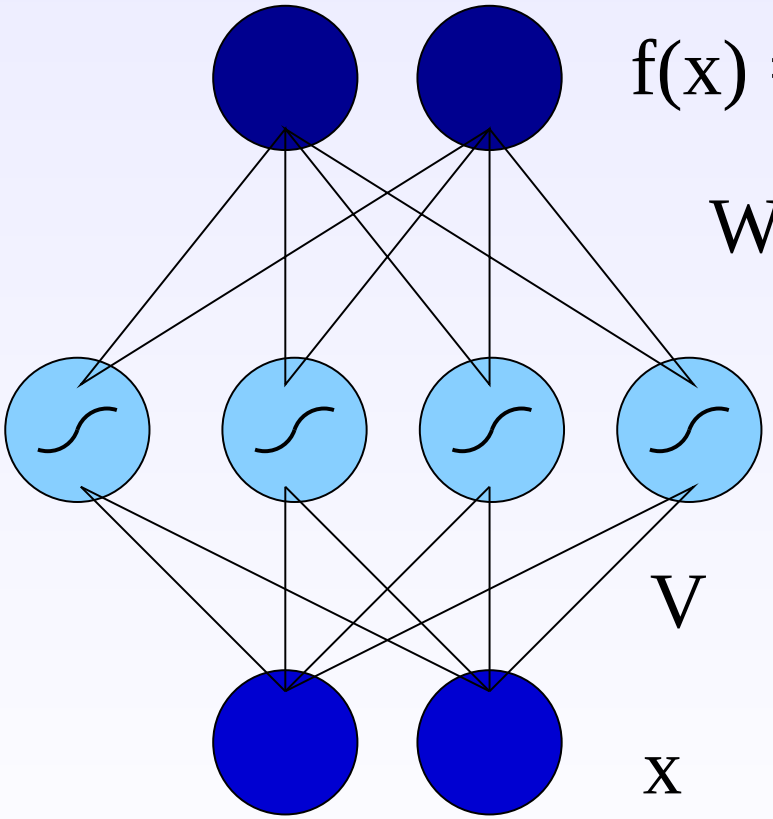
- Un réseau de neurones (à une couche cachée) est simplement une fonction ayant la forme suivante :

$$f(x) = h(\, o(x)\,) \qquad \text{avec} \qquad o_i(x) = \sum_j W_{ij} a_j(x)$$

$$a_j(x) = g(\sum_k V_{jk} x_k)$$

  où $g(\cdot)$ et $h(\cdot)$ sont des fonctions d'activation (sigmoid, tanh, softmax)
- Les réseaux de neurones peuvent approximer la plupart des fonctions, donc peuvent résoudre la plupart des tâches
- Tout ce qui manque, c'est un moyen d'obtenir les matrices W et V
- On appelle *algorithme d'apprentissage* une procédure qui accomplit cette tâche

Illustration



$f(x) = h(W\ g(V\ x)\ )$

W

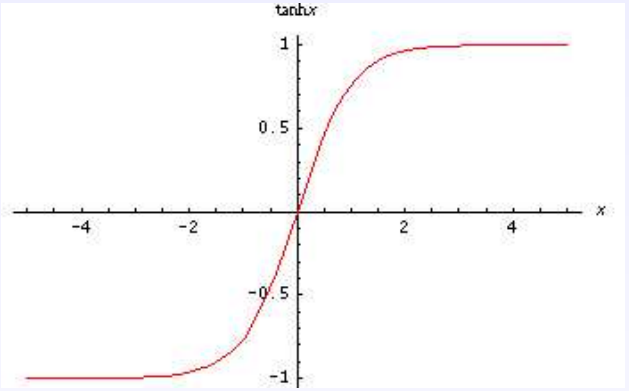
$g(V\ x)$

V

x

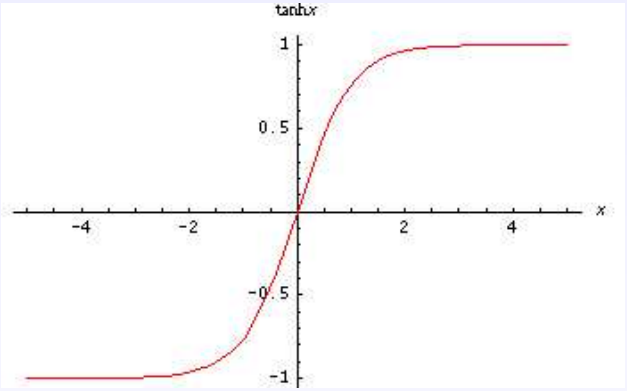
Fonctions d'activations

- Pour la couche cachée, la fonction d'activation sera la tangente hyperbolique (notée *tanh*) :



Fonctions d'activations

- Pour la couche cachée, la fonction d'activation sera la tangente hyperbolique (notée *tanh*) :



- Pour la couche de sortie, on utilisera la fonction softmax :

$$h(o)_i = \frac{e^{o_i}}{\sum_j e^{o_j}}$$

et on pourra alors noter $f(x_t)_{y_t}$ comme étant la probabilité que y_t soit associé à x_t selon notre modèle

Algorithme d'apprentissage

- Il existe différents algorithmes d'apprentissage pour les réseaux de neurones, tous avec leurs avantages et inconvénients

Algorithme d'apprentissage

- Il existe différents algorithmes d'apprentissage pour les réseaux de neurones, tous avec leurs avantages et inconvénients
- Dans ce cas-ci, on va utiliser la *descente de gradient stochastique*

Algorithme d'apprentissage

- Il existe différents algorithmes d'apprentissage pour les réseaux de neurones, tous avec leurs avantages et inconvénients
- Dans ce cas-ci, on va utiliser la *descente de gradient stochastique*
- On a besoin d'un *ensemble d'apprentissage* $\mathcal{T} = \{(x_t, y_t)\}_{t=1}^{n}$, qui correspond à un ensemble de paires

Algorithme d'apprentissage

- Il existe différents algorithmes d'apprentissage pour les réseaux de neurones, tous avec leurs avantages et inconvénients
- Dans ce cas-ci, on va utiliser la *descente de gradient stochastique*
- On a besoin d'un *ensemble d'apprentissage* $\mathcal{T} = \{(x_t, y_t)\}_{t=1}^{n}$, qui correspond à un ensemble de paires
- On a aussi besoin d'un coût à optimiser, qui ici correspondra à la log-vraisemblance négative des données, i.e. :

$$C(x_t, y_t, f(\cdot)) = -\log P(y_t|x_t, f(\cdot)) = -\log f(x_t)_{y_t}$$

Algorithme d'apprentissage

L'algorithme consiste en la procédure suivante :

1. Initialiser W et V
2. De $t = 1$ à $t = |\mathcal{T}|$
   1. calculer $C(x_t, y_t, f(\cdot))$
   2. mettre à jour $W_{ij} \leftarrow W_{ij} - \lambda \frac{\partial}{\partial W_{ij}} C(x_t, y_t, f(\cdot)) \forall i, j$
   3. mettre à jour $V_{jk} \leftarrow V_{jk} - \lambda \frac{\partial}{\partial V_{jk}} C(x_t, y_t, f(\cdot)) \forall j, k$
3. si le critère d'arrêt n'est pas satisfait, retourner à 2

Comme critère d'arrêt, on va utiliser surveiller la progression du coût sur un autre ensemble de pairs (x_t, y_t), appelé ensemble de validation, et on va cesser l'apprentissage lorsque ce coût augmentera.

Plan

1. Introduction
2. Réseaux de Neurones
3. Représentations Distribuées
4. Applications
   - Modélisation de la langue
   - Reconnaissance de la parole
   - Recherche d'Information
   - Classification syntaxique
   - Autres
5. Conclusion

Defi nition

- Une représentation distribuée d'un objet (e.g. mot) est simplement un vecteur de valeur réelle, de taille fixe D.

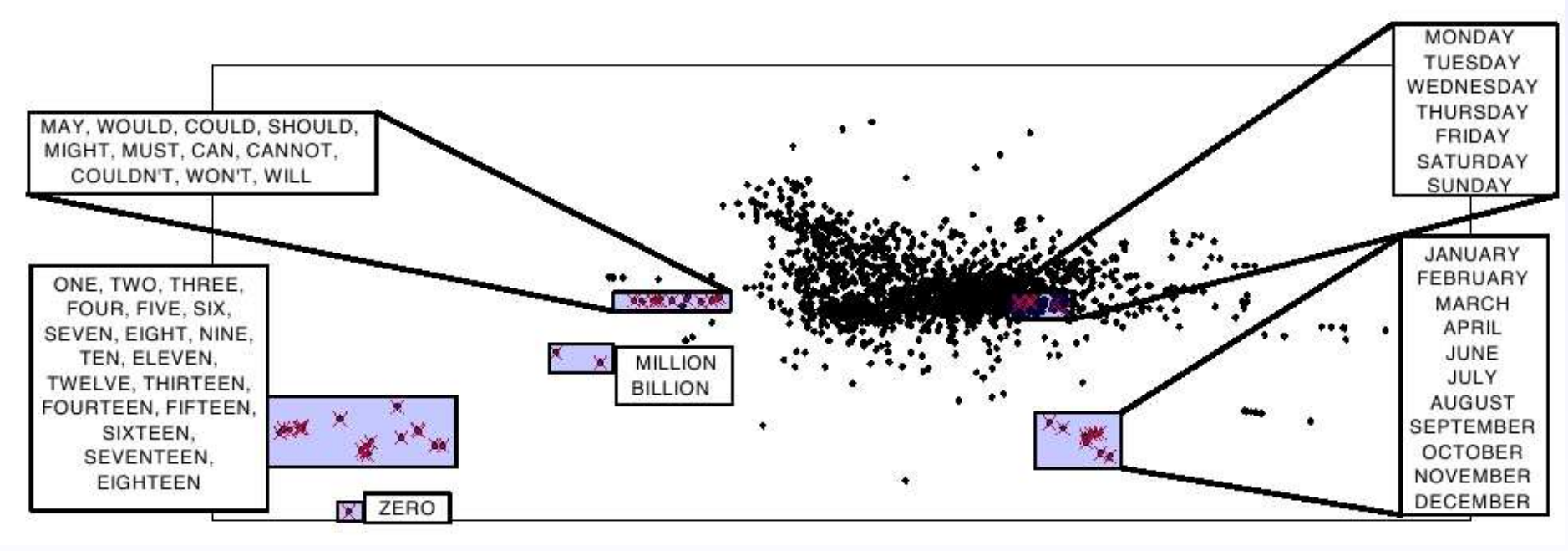
Defi nition

- Une représentation distribuée d'un objet (e.g. mot) est simplement un vecteur de valeur réelle, de taille fixe D.
- Les représentations sont normalement stockées dans une table C :

Mots w	Représentations Distribuées $C(w)$
"le"	[0.6762 -0.9607 0.3626 -0.2410 0.6636]
"la"	[0.6859 -0.9266 0.3777 -0.2140 0.6711]
"."	[-0.0069 0.7995 0.6433 0.2898 0.6359]
","	[0.0295 0.8150 0.6852 0.3182 0.6545]
"chien"	[0.5896 0.9137 0.0452 0.7603 -0.6541]
"chat"	[0.5965 0.9143 0.0899 0.7702 -0.6392]
"be"	[0.1656 -0.1530 0.0310 -0.3321 -0.1342]
"have"	[0.1760 -0.1340 0.0702 -0.2981 -0.1111]
...	...

Example

Voici de quoi peut avoir l'air un espace de représentations distribuées en 2D :



Apprentissage

- Il existe plus d'un algorithme pour apprendre les représentations distribuées des mots d'un vocabulaire

Apprentissage

- Il existe plus d'un algorithme pour apprendre les représentations distribuées des mots d'un vocabulaire
- Une famille d'algorithme populaire correspond aux algorithmes de réduction de dimensionalité (Latent Semantic Analysis (Deerwester et al., 1990), Semidefinite Embedding (Weinberger, Sha and Saul, 2004), etc)

Apprentissage

- Il existe plus d'un algorithme pour apprendre les représentations distribuées des mots d'un vocabulaire
- Une famille d'algorithme populaire correspond aux algorithmes de réduction de dimensionalité (Latent Semantic Analysis (Deerwester et al., 1990), Semidefinite Embedding (Weinberger, Sha and Saul, 2004), etc)
- Ici, on va plutôt utiliser les réseaux de neurones pour apprendre ces représentations

Apprentissage

- Il existe plus d'un algorithme pour apprendre les représentations distribuées des mots d'un vocabulaire
- Une famille d'algorithme populaire correspond aux algorithmes de réduction de dimensionalité (Latent Semantic Analysis (Deerwester et al., 1990), Semidefinite Embedding (Weinberger, Sha and Saul, 2004), etc)
- Ici, on va plutôt utiliser les réseaux de neurones pour apprendre ces représentations
- Cette façon de procéder a l'avantage de favoriser des représentations qui sont **utiles** pour la tâche considérée

Apprentissage

- L'idée est très simple : fixer les représentations des mots en entrée et faire l'apprentissage sur les poids du réseau ET sur les représentations

Apprentissage

- L'idée est très simple : fixer les représentations des mots en entrée et faire l'apprentissage sur les poids du réseau ET sur les représentations
- Donc, disons qu'on a en entrée une séquence de mots w_1^N, alors l'entrée x devient :

$$x = [C(w_1)C(w_2)\dots C(w_n)]$$

Apprentissage

- L'idée est très simple : fixer les représentations des mots en entrée et faire l'apprentissage sur les poids du réseau ET sur les représentations
- Donc, disons qu'on a en entrée une séquence de mots w_1^N, alors l'entrée x devient :

$$x = [C(w_1)C(w_2)\dots C(w_n)]$$

- Ainsi, une coordonnée x_i de l'entrée correspond vraiment à un champ C_{lm} de la table des représentations

Apprentissage

L'algorithme de descente de gradient stochastique devient donc :

1. Initialiser W, V et C
2. De $t = 1$ à $t = |T|$
   1. calculer $C(x_t, y_t, f(\cdot))$
   2. mettre à jour $W_{ij} \leftarrow W_{ij} - \lambda \frac{\partial}{\partial W_{ij}} C(x_t, y_t, f(\cdot)) \forall i, j$
   3. mettre à jour $V_{jk} \leftarrow V_{jk} - \lambda \frac{\partial}{\partial V_{jk}} C(x_t, y_t, f(\cdot)) \forall j, k$
   4. mettre à jour $x_{ti} = C_{lm} \leftarrow C_{lm} - \lambda \frac{\partial}{\partial C_{lm}} C(x_t, y_t, f(\cdot)) \forall i$
3. si le critère d'arrêt n'est pas satisfait, retourner à 2

Plan

1 Introduction

2 Réseaux de Neurones

3 Représentations Distribuées

4 Applications
- Modélisation de la langue
- Reconnaissance de la parole
- Recherche d'Information
- Classification syntaxique
- Autres

5 Conclusion

Neural Probabilistic Language Model

- Introduit par (Bengio, Ducharme and Vincent, 2001), ce modèle a popularisé les réseaux de neurones en traitement de la langue

Neural Probabilistic Language Model

- Introduit par (Bengio, Ducharme and Vincent, 2001), ce modèle a popularisé les réseaux de neurones en traitement de la langue
- La tâche correspond à :

$$w_{t-N+1}^{t-1} \rightarrow w_t$$

Neural Probabilistic Language Model

- Introduit par (Bengio, Ducharme and Vincent, 2001), ce modèle a popularisé les réseaux de neurones en traitement de la langue
- La tâche correspond à :

$$w_{t-N+1}^{t-1} \rightarrow w_t$$

- Ajout de liens directs entre la couche d'entrée et de sortie

Neural Probabilistic Language Model

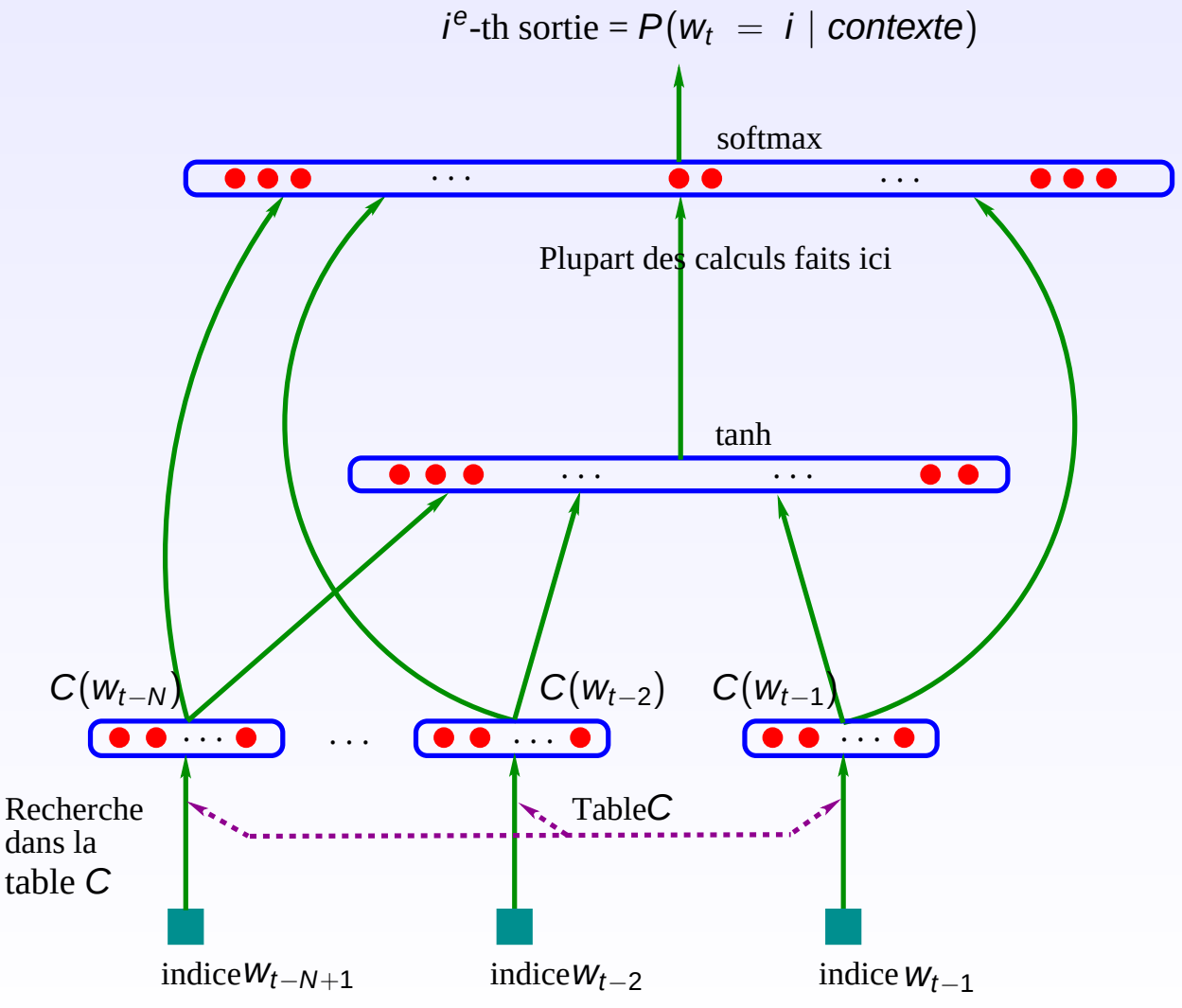
- Introduit par (Bengio, Ducharme and Vincent, 2001), ce modèle a popularisé les réseaux de neurones en traitement de la langue
- La tâche correspond à :

$$w_{t-N+1}^{t-1} \rightarrow w_t$$

- Ajout de liens directs entre la couche d'entrée et de sortie
- Résultats comparatif (perplexité) :

Modèle	Brown (1M)	AP (14M)	Temps
Kneser-Ney	321	117	"full vite"
NNet	259	.	"full long"
NNet interp.	**252**	**109**	"full long itout"

Neural Probabilistic Language Model



Accélération

- Utilisation d'une "short list" avec repli

Accélération

- Utilisation d'une "short list" avec repli
- Approximation du gradient à l'aide d'une estimation par "Importance Sampling" (Bengio and Senécal, 2003)

Accélération

- Utilisation d'une "short list" avec repli
- Approximation du gradient à l'aide d'une estimation par "Importance Sampling" (Bengio and Senécal, 2003)
- Utilisation d'une hiérarchie prédéfinie pour factoriser le calcul (Morin and Bengio, 2005)

$$P(w_t|w_{t-N+1}^{t-1}) = \prod_{t_i \in T(w_t)} P(t_i|t_{i-1}, w_{t-N+1}^{t-1})$$

Résultats

Voici les résultats obtenus sur le corpus Brown

Modèle	Perplexité	Temps Ent. / exemples	Temps Exec. / exemples
trigram	268.7		
class-based	249.1		
NNet	195.3	462.6 ms	270.7 s
NNet Sampling	192.6	6.73 ms	221.3 s
NNet Hiér.	220.7	1.79 ms	1.4 s

Extension

- Utilisation d'un réseau de neurones dans un "Structural Language Model" (Xu, Emami and Jelinek, 2003)

Extension

- Utilisation d'un réseau de neurones dans un "Structural Language Model" (Xu, Emami and Jelinek, 2003)
- L'idée, c'est que toutes les composantes du modèle correspondent à un réseau de neurones

Extension

- Utilisation d'un réseau de neurones dans un "Structural Language Model" (Xu, Emami and Jelinek, 2003)
- L'idée, c'est que toutes les composantes du modèle correspondent à un réseau de neurones
- Résultats (perplexité)

Modèle		+3gram
Kneser Ney	139.4	129.0
NNet	151.2	118.4

Extension

- Utilisation d'un réseau de neurones dans un "Structural Language Model" (Xu, Emami and Jelinek, 2003)
- L'idée, c'est que toutes les composantes du modèle correspondent à un réseau de neurones
- Résultats (perplexité)

Modèle		+3gram
Kneser Ney	139.4	129.0
NNet	151.2	118.4

- Le modèle avec réseau de neurones semble donc faire des "erreurs" différentes de celles d'un modèle ngram standard

Extension

- Utilisation d'un réseau de neurones dans un "Structural Language Model" (Xu, Emami and Jelinek, 2003)
- L'idée, c'est que toutes les composantes du modèle correspondent à un réseau de neurones
- Résultats (perplexité)

Modèle		+3gram
Kneser Ney	139.4	129.0
NNet	151.2	118.4

- Le modèle avec réseau de neurones semble donc faire des "erreurs" différentes de celles d'un modèle ngram standard
- Ce modèle peut être utilisé par l'algorithme EM

Approche

- Dans ce cas, le problème majeur est la vitesse d'exécution du modèle

Approche

- Dans ce cas, le problème majeur est la vitesse d'exécution du modèle
- Plusieurs trucs sont utilisés dans (Schwenk and Gauvain, 2005)
  1. Utilisation d'une "short list"
  2. Les exemples sont appris en block ("batch") pour profi ter des opérations de multiplication de matrices rapides
  3. Utilisation des libraires BLAS appropriées
  4. Mise à jour des probabilités du treillis de façon effi cace

Approche

- Dans ce cas, le problème majeur est la vitesse d'exécution du modèle
- Plusieurs trucs sont utilisés dans (Schwenk and Gauvain, 2005)
  1. Utilisation d'une "short list"
  2. Les exemples sont appris en block ("batch") pour profi ter des opérations de multiplication de matrices rapides
  3. Utilisation des libraires BLAS appropriées
  4. Mise à jour des probabilités du treillis de façon effi cace
- Tout ça prend $0.05 \times RT$

Résultats

- Le réseau de neurones est utilisé après un décodeur de parole standard, qui prend $0.95 \times RT$

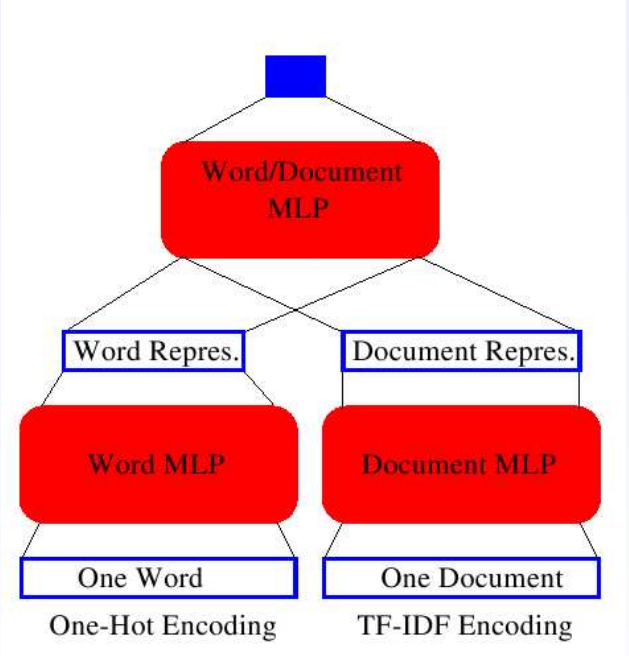
Résultats

- Le réseau de neurones est utilisé après un décodeur de parole standard, qui prend $0.95 \times RT$
- Résultats :

Modèle	Taille Entraînement	WER
Kneser-Ney	600M	14.24%
NNet interp.	4M	14.02%
NNet interp.	22M	13.88%
NNet interp.	92.5M	13.81%
NNet interp.	600M	13.75%

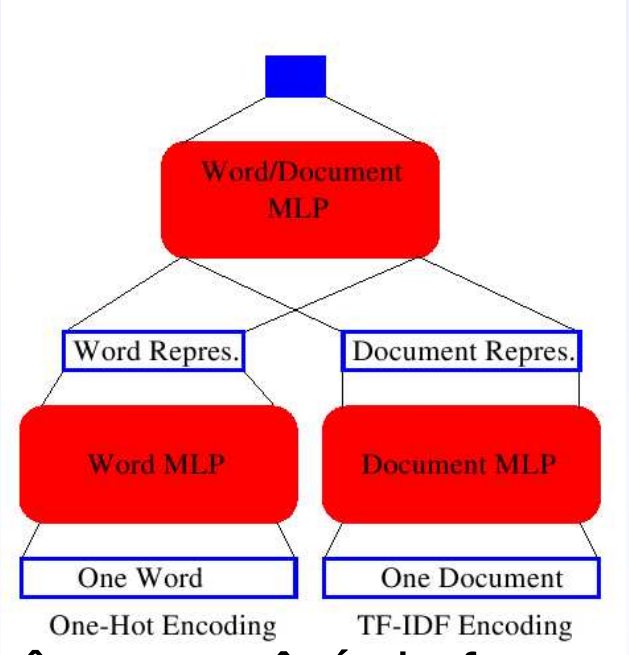
Neural Network for Text Representation

- (Keller and Bengio, 2005) propose réseau suivant pour comparer un mot d'une requête à un document :



Neural Network for Text Representation

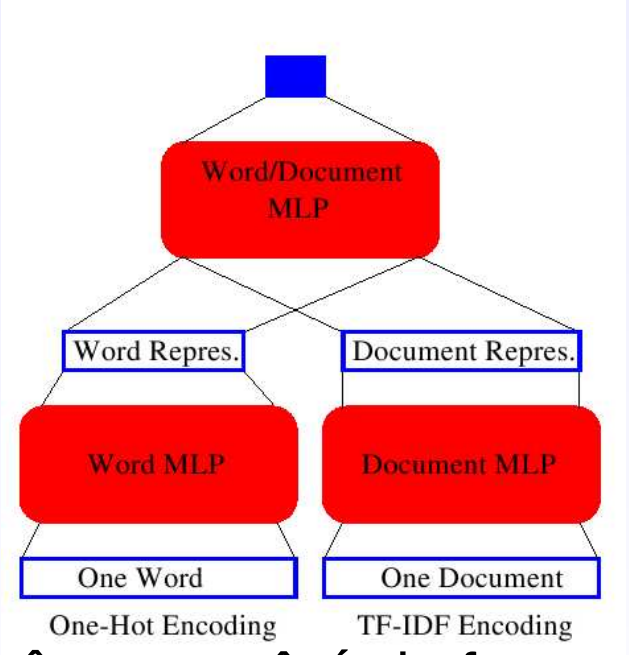
- (Keller and Bengio, 2005) propose réseau suivant pour comparer un mot d'une requête à un document :



- Ce réseau peut être entraîné de façon **non-supervisé**

Neural Network for Text Representation

- (Keller and Bengio, 2005) propose réseau suivant pour comparer un mot d'une requête à un document :



- Ce réseau peut être entraîné de façon **non-supervisé**
- Le critère d'entraînement est légèrement différent, pour compenser le déséquilibre entre les exemples positifs et négatifs

Résultats

- La distance entre une requête et un document est la somme normalisée des distances entre les mots de la requête et le document

Résultats

- La distance entre une requête et un document est la somme normalisée des distances entre les mots de la requête et le document
- Cette distance est interpolée avec celle donnée par un modèle TFIDF

Résultats

- La distance entre une requête et un document est la somme normalisée des distances entre les mots de la requête et le document
- Cette distance est interpolée avec celle donnée par un modèle TFIDF
- Résultats sur TDT2, avec requêtes TREC-9 (précision moyenne)

Modèle	"Document Retrieval"	"Batch Filtering"
TFIDF	0.170	0.185
PLSA	0.199	0.189
NNet	0.215	0.192

Généralisation à de nouveaux mots

- Parce que les représentations sont apprises pour un vocabulaire prédéterminé, on ne peut pas les généraliser directement à de nouveaux mots

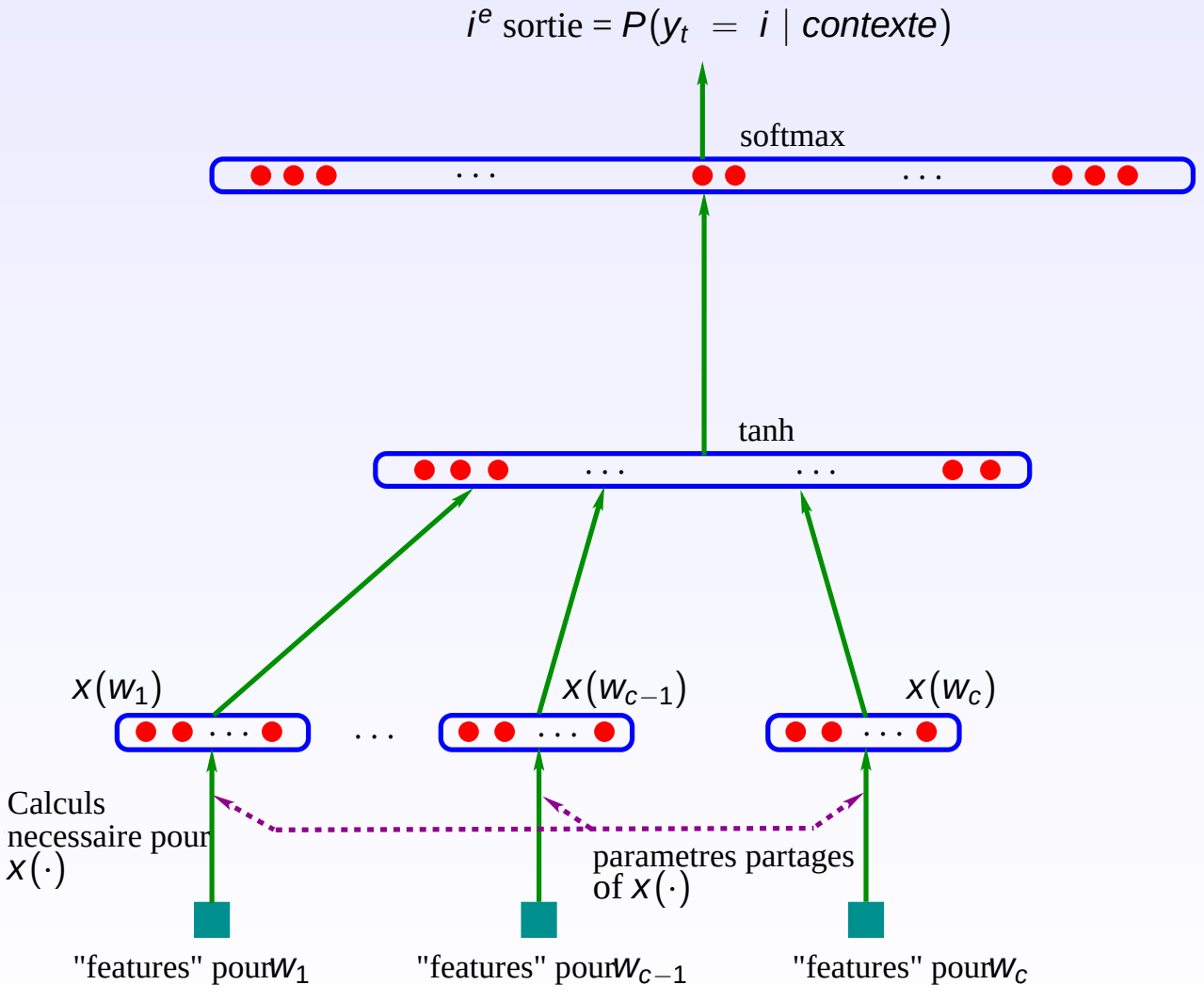
Généralisation à de nouveaux mots

- Parce que les représentations sont apprises pour un vocabulaire prédéterminé, on ne peut pas les généraliser directement à de nouveaux mots
- On pourrait prendre une moyenne pondérée des représentations déjà calculées, mais rien ne garantit que cette représentation sera utile

Généralisation à de nouveaux mots

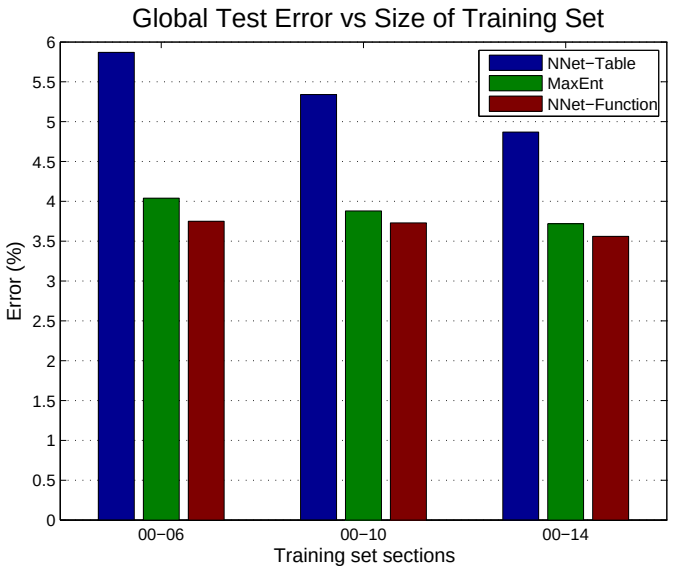
- Parce que les représentations sont apprises pour un vocabulaire prédéterminé, on ne peut pas les généraliser directement à de nouveaux mots
- On pourrait prendre une moyenne pondérée des représentations déjà calculées, mais rien ne garantit que cette représentation sera utile
- Ce problème est crucial, entre autre pour la classification syntaxique

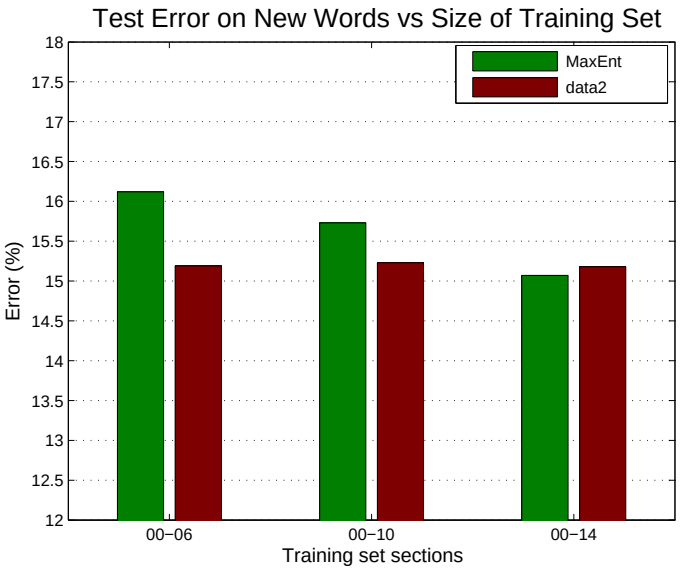
Nouvelle architecture



Résultats

Modèle	Erreur totale	Erreur sur les nouveaux
NNet-Table	4.49%	35.53%
MaxEnt	3.47%	14.80%
NNet-Function	3.32%	14.21%





Représentations apprises

Mot	Plus proches voisins
will	could should would might must
the	any an The a An
and	but or nor & yet
it	he she him us me
Eight	Seven Twenty Four Nine Three
make	want take give allow reduce
was	got had did ran met
caused	allowed received requested pushed determined
more	closer earlier higher faster tougher
than	because after about from without
30	44 65 61 900 55
researchers	brothers scientists teachers philosophers
.	? !
–	; : ... - 'n'

Et ç a continue...

- Désambiguïsation de phrases prépositionnelles

Et ç a continue...

- Désambiguïsation de phrases prépositionnelles
- Système de recommandation de pièce musicale de radiolibre.ca

Et ç a continue...

- Désambiguïsation de phrases prépositionnelles
- Système de recommandation de pièce musicale de radiolibre.ca
- Et ça peut continuer encore ...

Plan

1. Introduction
2. Réseaux de Neurones
3. Représentations Distribuées
4. Applications
   - Modélisation de la langue
   - Reconnaissance de la parole
   - Recherche d'Information
   - Classification syntaxique
   - Autres
5. Conclusion

Ce qu'il reste à faire

- Appliquer à d'autres tâches (traduction automatique)

Ce qu'il reste à faire

- Appliquer à d'autres tâches (traduction automatique)
- Accélérer et faciliter l'entraînement dans le cas de la modélisation du langage

Ce qu'il reste à faire

- Appliquer à d'autres tâches (traduction automatique)
- Accélérer et faciliter l'entraînement dans le cas de la modélisation du langage
- Étudier l'impact de différentes "features" sur la prédiction des représentations et la performance du réseau de neurones

Ce qu'il reste à faire

- Appliquer à d'autres tâches (traduction automatique)
- Accélérer et faciliter l'entraînement dans le cas de la modélisation du langage
- Étudier l'impact de différentes "features" sur la prédiction des représentations et la performance du réseau de neurones
- Étudier l'impact de l'apprentissage mutli-tâche sur les représentations apprises

Liens

- Pour accéder à ces acétates, sur ma page web, section “Publications” ;

  `http://www.iro.umontreal.ca/~larocheh`

  ou

  Google : Hugo Larochelle

- Pour en connaître plus sur l’apprentissage machine, assistez aux séminaires UdeM-McGill-MITACS sur le sujet :

  `http://www.iro.umontreal.ca/~lisa`

  (section “Séminaires”)

  ou

  Google : Séminaires Apprentissage Machine

Bengio, Y., Ducharme, R., and Vincent, P. (2001).
A neural probabilistic language model.
In Leen, T. K., Dietterich, T. G., and Tresp, V., editors, *Advances in Neural Information Processing Systems 13*, pages 932–938. MIT Press.

Bengio, Y. and Senécal, J.-S. (2003).
Quick training of probabilistic neural nets by importance sampling.
In *Proceedings of AISTATS'2003*.

Deerwester, S., Dumais, S., Furnas, G., Landauer, T., and Harshman, R. (1990).
Indexing by latent semantic analysis.
Journal of the American Society for Information Science, 41(6) :391–407.

Keller, M. and Bengio, S. (2005).

A neural network for text representation.
In *Proceedings of the 15th International Conference on Artificial Neural Networks : Biological Inspirations, ICANN, Lecture Notes in Computer Science*, volume LNCS 3697, pages 667–672.

Morin, F. and Bengio, Y. (2005).
Hierarchical probabilistic neural network language model.
In *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics, Jan 6-8, 2005, Savannah Hotel, Barbados*.

Schwenk, H. and Gauvain, J.-L. (2005).
Training Neural Network Language Models On Very Large Corpora.
In *Joint Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 201–208, Vancouver.

Sutton, C. and McCallum, A. (2006).
An introduction to conditional random fields for relational learning.
In Getoor, L. and Taskar, B., editors, *Introduction to Statistical Relational Learning*. MIT Press.
To appear.

Weinberger, K. Q., Sha, F., and Saul, L. K. (2004).
Learning a kernel matrix for nonlinear dimensionality reduction.
In *Proceedings of the Twenty First International Conference on Machine Learning (ICML-04)*, pages 839–846, Banff, Canada.

Xu, P., Emami, A., and Jelinek, F. (2003).
Training connectionist models for the structured language model.
In *Proceedings of EMNLP*.