TÂCHES ET MÉTRIQUES

Language Reading Group

June 20, 2019

Étienne

LIP6 - Sorbonne Université

Georgetown–IBM experiment (1954)
Traduction russe → anglais de **60** phrases.

- 250 lexical items
- 6 grammar rules

Georgetown–IBM experiment (1954)
Traduction russe → anglais de **60** phrases.

- 250 lexical items
- 6 grammar rules

Five, perhaps three years hence, interlingual meaning conversion by electronic process in important functional areas of several languages may well be an accomplished fact. — Leon Dostert

Georgetown–IBM experiment (1954)
Traduction russe → anglais de **60** phrases.

- 250 lexical items
- 6 grammar rules

Five, perhaps three years hence, interlingual meaning conversion by electronic process in important functional areas of several languages may well be an accomplished fact. — Leon Dostert

It's true there's been a lot of work on trying to apply statistical models to various linguistic problems. I think there have been some successes, but a lot of failures. There is a notion of success ... which I think is novel in the history of science. It interprets success as approximating unanalyzed data. — Noam Chomsky (2011)
`http://norvig.com/chomsky.html`

Tâches

Tâches principales

Traitement automatique du langage naturelle $\Rightarrow$ langage naturelle

Traitement automatique du langage naturelle $\Rightarrow$ langage naturelle

Language Modeling

INPUT: **Langage**

p("Pytorch is overrated.") = beaucoup

p("Colorless green ideas sleep furiously.") = pas beaucoup

Traitement automatique du langage naturelle $\Rightarrow$ langage naturelle

Language Modeling

INPUT: **Langage**
p("Pytorch is overrated.") = beaucoup
p("Colorless green ideas sleep furiously.") = pas beaucoup

Machine Translation

INPUT: **Langage** OUTPUT: **Langage**
Ça va. $\rightarrow$ It goes.

Traitement automatique du langage naturelle ⇒ langage naturelle

Language Modeling

INPUT: **Langage**
p("Pytorch is overrated.") = beaucoup
p("Colorless green ideas sleep furiously.") = pas beaucoup

Machine Translation

INPUT: **Langage** OUTPUT: **Langage**
Ça va. → It goes.

Question Answering

INPUT: **Langage**, ~**Langage** OUTPUT: ~**Langage**
Query + Connaissance → Réponse

Natural Language Understanding

INPUT: **Langage** OUTPUT: Données structurées

Paris is the capital of France. $\rightarrow$ CAPITAL(FRANCE) = PARIS

Natural Language Understanding

INPUT: **Langage** OUTPUT: Données structurées

Paris is the capital of France. $\rightarrow$ CAPITAL(FRANCE) = PARIS

Natural Language Generation

INPUT: Données structurées OUTPUT: **Langage**

CAPITAL(FRANCE) = PARIS $\rightarrow$ Paris is the capital of France.

Natural Language Understanding

INPUT: **Langage** OUTPUT: Données structurées

Paris is the capital of France. $\rightarrow$ CAPITAL(FRANCE) = PARIS

Natural Language Generation

INPUT: Données structurées OUTPUT: **Langage**

CAPITAL(FRANCE) = PARIS $\rightarrow$ Paris is the capital of France.

Named-Entity Recognition

INPUT: **Langage** OUTPUT: Named-Entity Tags

$[\text{Jim}]_{\text{Person}}$ bought 300 shares of $[\text{Acme}]_{\text{Organization}}$ in $[\text{2006}]_{\text{Time}}$.

Natural Language Understanding

INPUT: **Langage** OUTPUT: Données structurées

Paris is the capital of France. $\rightarrow$ CAPITAL(FRANCE) = PARIS

Natural Language Generation

INPUT: Données structurées OUTPUT: **Langage**

CAPITAL(FRANCE) = PARIS $\rightarrow$ Paris is the capital of France.

Named-Entity Recognition

INPUT: **Langage** OUTPUT: Named-Entity Tags

$[\text{Jim}]_{\text{Person}}$ bought 300 shares of $[\text{Acme}]_{\text{Organization}}$ in $[\text{2006}]_{\text{Time}}$.

Part-Of-Speech Tagging (étiquetage morpho-syntaxique)

INPUT: **Langage** OUTPUT: POS Tags

$\text{Nous}_{\text{PRO:PER}}$ $\text{sommes}_{\text{VER:pres}}$ $\text{allées}_{\text{VER:pper}}$ en_{PRP} $\text{Bretagne}_{\text{NAM}}$ $\cdot_{\text{SENT}}$

Topic Modeling

INPUT: **Langage** OUTPUT: Topic

I do not dislike Legally Blonde. $\rightarrow$ movie

Topic Modeling

INPUT: **Langage** OUTPUT: Topic
I do not dislike Legally Blonde. → movie

Sentiment Analysis

INPUT: **Langage** OUTPUT: Polarité
I do not dislike Legally Blonde. → positif

Topic Modeling

INPUT: **Langage** OUTPUT: Topic
I do not dislike Legally Blonde. → movie

Sentiment Analysis

INPUT: **Langage** OUTPUT: Polarité
I do not dislike Legally Blonde. → positif

Textual Entailment

INPUT: **Langage**, **Langage** OUTPUT: Relation logique
If you help the needy, God will reward you.
Giving money to a poor man has no consequences.
→ negative entailment (contradiction)

Handwriting recognition

INPUT: Image OUTPUT: **Langage**

Hypothèse : on écrit pas une suite de caractère aléatoire.

Tâches hybrides

Handwriting recognition

INPUT: Image OUTPUT: **Langage**

Hypothèse : on écrit pas une suite de caractère aléatoire.

Image captioning

INPUT: Image OUTPUT: **Langage**

Décrire une image en ∼une phrase.

Handwriting recognition

INPUT: Image OUTPUT: **Langage**

Hypothèse : on écrit pas une suite de caractère aléatoire.

Image captioning

INPUT: Image OUTPUT: **Langage**

Décrire une image en ~une phrase.

Text to speech

INPUT: **Langage** OUTPUT: Audio

Génération d'audio.

Métriques

Negative Log Likelihood

Habituellement on optimise la NLL:

$$\text{Loss}(\text{mot}) = -\log p(\text{mot}|\text{paramètres})$$

Minimiser ça $\rightarrow$ augmenter la probabilité que le modèle sorte le bon mot.

Negative Log Likelihood

Habituellement on optimise la NLL:

$$\text{Loss}(\text{mot}) = -\log p(\text{mot}|\text{paramètres})$$

Minimiser ça $\rightarrow$ augmenter la probabilité que le modèle sorte le bon mot.

En **sommant** sur la longueur de la phrase:

$$\text{Loss}(\text{phrase}) = \sum_{\substack{\text{pour chaque mot} \\ \text{dans la phrase}}} -\log p(\text{mot}|\text{paramètres})$$

Soit $p(x)$ la distribution empirique des mots (1 pour le bon mot, 0 pour les autres) et $q(x)$ la distribution prédite (la sortie de votre softmax).
Idéalement $q(x) = p(x)$.
On minimise donc la Kullback–Leibler divergence entre p et q.

$$D_{\mathrm{KL}}(p, q) = H(p, q) - H(p) = H(p, q)$$

$H(p, q)$ est la cross-entropy entre p et q c'est le nombre de bits qu'il faut utiliser pour coder un élément de p en se basant sur q.

$$H(p, q) = - \sum_{\substack{\text{pour chaque mot x} \\ \text{du vocabulaire}}} p(x) \log q(x)$$

$$H(p, q) = - \log q(\text{le bon mot de p})$$

La loss dépend de la longueur de la phrase et est difficile à interpréter, on utilise au lieu la perplexité :

$$2^{H(p,q)} = 2^{-\frac{1}{|dataset|}\sum_{x \in \text{dataset}} p(x) log_2 q(x)}$$

C'est l'exponentielle de la moyenne des cross-entropy sur les mots.

Un peu moins pourri que la perplexité.
Une généralisation de la précision : est ce que les mots que j'ai générés sont dans la/les référence(s).

$$\text{mprecision}_1 = \frac{\text{nombre de mot prédit qui sont dans la ref (clipped)}}{\text{longueur prediction}}$$

Brevity penalty: Si la prédiction est plus courte que la reference, on multiplie par une pénalité.

$$BP = \max\left(1, \exp\left(1 - \frac{\text{longueur reference}}{\text{longueur prediction}}\right)\right)$$

$$\text{BLEU} = \text{BP} \times \exp\left(\sum_{n=1}^{4} \frac{1}{4} \log \text{mprecision}_n\right)$$