Traitement automatique du langage

Méthodes statistiques, apprentissage profond, Transformers et LLM

Redha Moulla

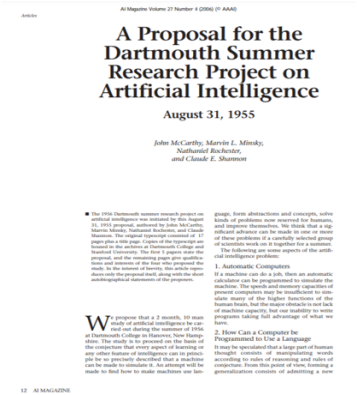
Plan

PARTIE I

Fondements et données textuelles

1956 : atelier de Dartmouth

L'histoire du NLP est étroitement liée à celle de l'intelligence artificielle. Le projet de recherche de Dartmouth mentionne notamment le langage et la traduction ; cet atelier est souvent considéré comme l'un des actes fondateurs du champ de l'IA.



AI Magazine Volume 27 Number 4 (2006) (© AAAI)

Articles

A Proposal for the Dartmouth Summer Research Project on Artificial Intelligence

August 31, 1955

John McCarthy, Marvin L. Minsky, Nathaniel Rochester, and Claude E. Shannon

■ The 1956 Dartmouth summer research project on artificial intelligence was initiated by this August 31, 1955 proposal, authored by John McCarthy, Marvin Minsky, Nathaniel Rochester, and Claude Shannon. The original typescript consisted of 17 pages plus a title page. Copies of the typescript are housed in the archives at Dartmouth College and Stanford University. The first 5 pages state the proposal, and the remaining pages give qualifications and interests of the four participants who would study. In the interest of brevity, this article reproduces only the proposal itself, along with the short autobiographical statements of the proposers.

We propose that a 2 month, 10 man study of artificial intelligence be carried out during the summer of 1956 at Dartmouth College in Hanover, New Hampshire. The study is to proceed on the basis of the conjecture that every aspect of learning or any other feature of intelligence can in principle be so precisely described that a machine can be made to simulate it. An attempt will be made to find how to make machines use language, form abstractions and concepts, solve kinds of problems now reserved for humans, and improve themselves. We think that a significant advance can be made in one or more of these problems if a carefully selected group of scientists work on it together for a summer.

The following are some aspects of the artificial intelligence problem:

1. Automatic Computers

If a machine can do a job, then an automatic calculator can be programmed to simulate the machine. The speeds and memory capacities of present computers may be insufficient to simulate many of the higher functions of the human brain, but the major limitation is not lack of machine capacity, but our inability to write programs taking full advantage of what we have.

2. How Can a Computer be Programmed to Use a Language

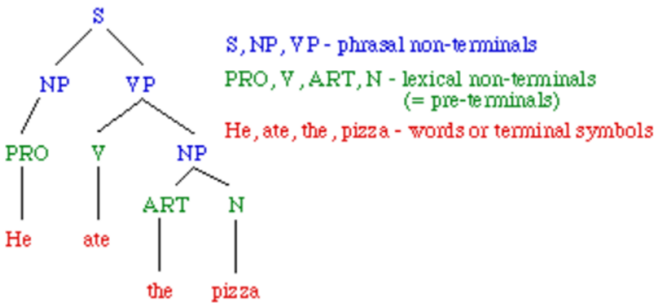
It may be speculated that a large part of human thought consists of manipulating words according to rules of reasoning and rules of conjecture. From this point of view, forming a generalization consists of admitting a new

12 AI MAGAZINE

"We propose that a 2-month, 10-man study of artificial intelligence be carried out during the summer of 1956 at Dartmouth College in Hanover, New Hampshire. *The study is to proceed on the basis of the conjecture that every aspect of learning or any other feature of intelligence can in principle be so precisely described that a machine can be made to simulate it. An attempt will be made to find how to make machines use language, form abstractions and concepts, solve kinds of problems now reserved for humans, and improve themselves. We think that a significant advance can be made in one or more of these problems if a carefully selected group of scientists work on it together for a summer."*

Années 1970 : approches symboliques

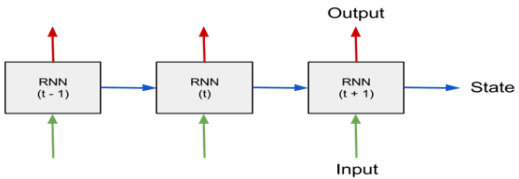
Une grande partie des systèmes de l'époque combine grammaires formelles, lexiques, règles et analyse syntaxique. Ces méthodes rendent les connaissances explicites, mais leur couverture et leur maintenance deviennent difficiles lorsque les domaines et les usages se diversifient.



Années 1980 : réseaux récurrents

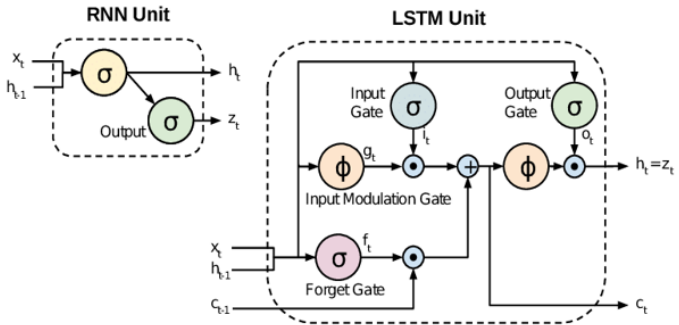
Les réseaux récurrents partagent leurs paramètres dans le temps et construisent un état dépendant du passé. Les architectures de Jordan et d'Elman ont popularisé ce cadre, sensible aux gradients évanescents ou explosifs.

We describe a new learning procedure, back-propagation, for networks of neurone-like units. The procedure repeatedly adjusts the weights of the connections in the network so as to minimize a measure of the difference between the actual output vector of the net and the desired output vector. As a result of the weight adjustments, internal 'hidden' units which are not part of the input or output come to represent important features of the task domain, and the regularities in the task are captured by the interactions of these units. The ability to create useful new features distinguishes back-propagation from earlier, simpler methods such as the perceptron-convergence procedure[1].



1997 : Long Short-Term Memory

Les LSTM sont conçus pour dépasser certaines limites des RNN, en particulier la disparition du gradient sur les dépendances longues.



Années 2010 : renaissance de l'apprentissage profond

Plusieurs avancées ont été réalisées durant la dernière décennie en apprentissage profond, notamment depuis 2012 (AlexNet).

- 2013 : Word2Vec popularise les embeddings prédictifs à grande échelle
- 2014 : attention neuronale dans les modèles encodeur–décodeur
- 2017 : Transformer et self-attention comme opérateur central
- 2018 : ELMo, BERT, GPT.
- 2020 : GPT-3.
- 2022 : ChatGPT, etc.

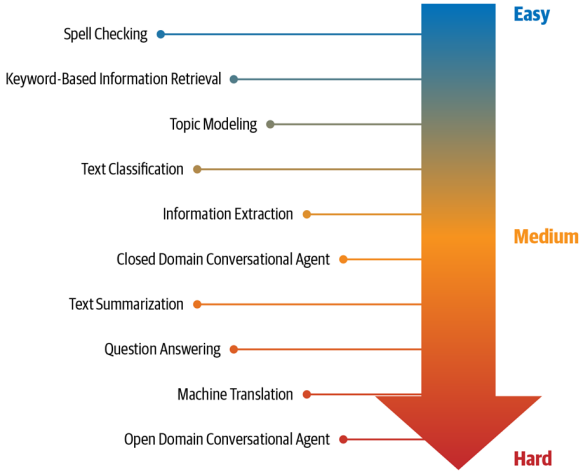
Tâches classiques en NLP

On retrouve en NLP un ensemble de tâches canoniques, utiles aussi bien dans les applications que pour comparer les méthodes.

- Modélisation du langage (*language modeling*)
- Classification de texte
- Extraction d'informations
- Détection de thématiques (*topic modeling*)
- Résumé de texte
- Question/réponse
- Traduction automatique
- Agents conversationnels (domaine ouvert ou fermé)

Classement des tâches en NLP par ordre de difficulté

Le classement ci-dessous est à titre indicatif et sujet à des changements. Certaines tâches, comme la traduction, paraissent aujourd'hui moins difficiles qu'il y a quelques années.



Pourquoi le NLP est si difficile ?

- Le langage naturel est **ambigu**
  - *"Acheter un avocat"*
- Le langage naturel repose sur le **sens commun**
  - Il fait froid *en hiver*
  - Le beurre fond dans le four
- Le langage ne se réduit pas à un petit ensemble de **règles déterministes et sans exceptions**
  - *En bleu adorable fleurit*
    Le toit de métal du clocher. Alentour
    Plane un cri d'hirondelles, autour
    S'étend le bleu le plus touchant. Le soleil
    Friedrich Hölderlin

Prétraitement des données textuelles

Les données textuelles sont faiblement structurées. Les opérations de prétraitement dépendent de la tâche et du modèle :

1. Segmentation des phrases et **tokenisation** ;
2. normalisation Unicode, nettoyage et contrôle de l'encodage ;
3. éventuellement suppression de *stop words*, stemming, lemmatisation ou passage en minuscules.

Pour les modèles préentraînés, une normalisation agressive peut supprimer des signaux utiles (casse, ponctuation, accents). Toute transformation doit donc être validée expérimentalement.

Tokenisation

La tokenisation consiste à segmenter une phrase en unités plus petites (*tokens*), qui sont généralement des mots, mais qui peuvent également être des caractères ou ensembles de caractères.

Exemple:

- "Il fait beau aujourd'hui à Paris" $\rightarrow$ ["Il", "fait", "beau", "aujourd'hui", "à", "Paris"]
- "Paris" $\rightarrow$ ["P", "a", "r", "i", "s"]

La tokenisation en sous-mots est la norme pour les Transformers ; le vocabulaire et l'algorithme de segmentation font partie intégrante du modèle.

Stemming et lemmatisation

- Le **stemming** applique des règles de troncature ; la forme obtenue n'est pas nécessairement un mot du lexique.
  - marcher → march
  - marchait → march
  - marcha → march
  - marchassiez → march
  - marché → march

- La **lemmatisation** ramène une forme fléchie à son lemme canonique en utilisant des informations morphologiques, lexicales et parfois contextuelles.
  - nous → nous
  - sommes → être
  - venus → venir
  - faire → faire
  - les → le
  - marchés → marché

Vectorisation

- **Encodage one-hot :** pour un vocabulaire de taille $|V|$, chaque mot est représenté par un vecteur binaire de dimension $|V|$.
  - $V = \{Tom, mange, pomme, chien, ciel\}$, où dim$(V) = 5$
  - Tom = [1, 0, 0, 0, 0]
  - mange = [0, 1, 0, 0, 0]
  - pomme = [0, 0, 1, 0, 0]
  - etc.

- **N-grams**
  - "machine learning est très utilisé en NLP."
  - 1-gramme : {machine, learning, est, très, utilisé, en, NLP}
  - 2-grammes : {machine learning, learning est, est très, très utilisé, utilisé en, en NLP}

Extraction d'informations

L'information peut concerner différentes entités (événements clés, personnes, lieux, etc.).
L'extraction d'information inclut :

- Détection d'entités nommées (NER).
- Extraction de mots clés (KPE).
- Extraction de relations.

Entités nommées (NER)

Entrée

Marie Curie rejoint Paris le 3 novembre 1891.

Mention	Type	Span
Marie Curie	PER	[0, 11[
Paris	LOC	[20, 25[
3 novembre 1891	DATE	[29, 44[

Formulation

- segmentation et typage conjoints ;
- schéma BIO/BILOU au niveau token, ou prédiction directe de spans ;
- catégories définies par une convention d'annotation, non par une ontologie universelle.

Évaluation

Exact-match sur type et frontières ; rapporter précision, rappel et F_1 par type, puis analyser les entités imbriquées et hors domaine.

Extraction de relations

Exemple

Ada Lovelace a collaboré avec Charles Babbage sur la machine analytique.

(A. Lovelace, `collabore`, C. Babbage)

(A. Lovelace, `travaille_sur`, machine analytique)

Variantes

- classification d'une paire d'entités ;
- extraction jointe entités–relations ;
- relations intra- ou inter-phrases ;
- relations n-aires, temporelles ou directionnelles.

Les « négatifs » sont rarement exhaustifs : absence d'annotation ne signifie pas toujours absence de relation.

Extraction de termes et phrases clés

Approches extractives

- filtrage morphosyntaxique de candidats ;
- TF–IDF, TextRank, similarité document–candidat ;
- séquence BIO ou classement supervisé.

Approches génératives

- peuvent produire une phrase absente du texte ;
- utiles pour normaliser synonymes et concepts ;
- plus difficiles à contraindre et à évaluer.

Évaluation

$P@k$, $R@k$, $F_1@k$, avec normalisation et appariement explicités. Une référence humaine est souvent incomplète et plusieurs sorties sont légitimes.

Étiquetage morphosyntaxique (POS)

Le	pilote	ferme	la	porte	.
DET	NOUN	VERB	DET	NOUN	PUNCT

Dans *la ferme*, *ferme* est un nom ; dans *il ferme*, c'est un verbe : l'étiquette dépend du contexte.

- tags universels ou jeu spécifique ;
- POS parfois enrichi de traits morphologiques ;
- décodage local ou structuré ;
- évaluer par étiquette, ambiguïté et domaine, pas seulement par exactitude globale.

Source : Universal Dependencies, schéma UPOS.

Pourquoi une tokenisation sous-lexicale ?

Problème d'ingénierie

Un vocabulaire de mots est ouvert : flexions, composés, fautes, identifiants et néologismes produisent des mots hors vocabulaire.

- **Caractères / octets** : vocabulaire compact, séquences longues.
- **Mots** : séquences courtes, vocabulaire énorme.
- **Sous-mots** : compromis appris sur le corpus.

Effet du vocabulaire V

paramètres de l'embedding $\simeq |V|d$

coût attention $\simeq O(n^2 d)$

Réduire n accélère l'attention, mais augmenter $|V|$ alourdit les projections d'entrée et de sortie.

La tokenisation fait partie du modèle : elle modifie coût, biais et robustesse.

Byte Pair Encoding : algorithme et exemple

Apprentissage du vocabulaire

1. Initialiser les unités par caractères ou octets.
2. Compter les paires adjacentes sur le corpus pondéré.
3. Fusionner la paire la plus fréquente.
4. Répéter jusqu'à la taille de vocabulaire cible.

Exemple simplifié

$\{\texttt{bas}, \texttt{basse}, \texttt{basses}\}$
`b+a` $\rightarrow$ `ba`, puis `ba+s` $\rightarrow$ `bas`, puis `s+e` $\rightarrow$ `se`.
`basses` $\mapsto$ `bas + se + s`.

À maîtriser

- apprentissage $\neq$ encodage ;
- les fusions sont ordonnées ;
- BPE n'est pas une analyse morphologique ;
- l'unité `token` n'est ni un mot ni nécessairement un caractère.

Source : Sennrich et al., 2016 ; Gage, 1994.

BPE, WordPiece, Unigram et tokenisation par octets

Méthode	Principe d'apprentissage	Conséquence pratique
BPE	fusions gloutonnes des paires fréquentes	déterministe, simple ; les fusions locales ne maximisent pas une vraisemblance globale
WordPiece	ajout de sous-mots améliorant un critère de corpus	segmentation proche de BPE, vocabulaire incluant des marqueurs de continuation
Unigram LM	part d'un grand lexique puis élague ; maximise une vraisemblance marginale	plusieurs segmentations possibles ; *subword regularization*
Octets / byte-level	alphabet de base couvrant tout UTF-8	aucun OOV ; séquences parfois longues et frontières linguistiques moins naturelles

Le choix se valide sur la langue, le domaine, la longueur en tokens, le taux de fragmentation et la tâche aval — pas seulement sur la taille du vocabulaire.

Source : Kudo & Richardson, 2018 ; Schuster & Nakajima, 2012.

Audit d'un tokeniseur pour le français

Échantillonner des cas difficiles

- apostrophes : *l'État*, *aujourd'hui* ;
- diacritiques et Unicode : *cœur*, caractères composés ;
- nombres, unités, dates, code, URLs ;
- jargon métier et noms propres ;
- variantes régionales et langues mêlées.

Mesurer

$$\rho = \frac{\#\text{tokens}}{\#\text{mots}}$$

$$f_{\geq k} = \Pr(\text{mot segmenté en } \geq k)$$

Comparer ces métriques par sous-population, puis vérifier l'impact sur la performance et la latence.

Remarque

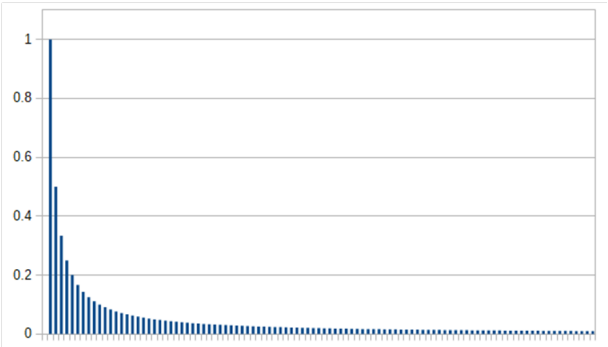
Normaliser les accents ou la casse peut détruire un signal utile et créer une divergence entre l'entraînement et l'évaluation.

PARTIE II

Méthodes statistiques et modèles classiques

Principe des techniques statistiques pour le NLP

- Un document est caractérisé par la distribution des mots qui le composent.
- La fréquence lexicale suit approximativement une loi de Zipf : une petite fraction du vocabulaire concentre une grande part des occurrences.



Fréquence du terme (TF)

- Term Frequency (TF): la fréquence avec laquelle un terme t apparaît dans un document d.

$$\mathrm{tf}(t,d) = \frac{f(t,d)}{\sum_{t'} f(t',d)}$$

Exemple : étant donné 1000 mots.

Terme	**Fréquence**	**TF score**
chat	15	0.015
lait	4	0.004
mange	6	0.006
feu	3	0.003
...	...	...

Fréquence documentaire inverse (IDF)

- L'Inverse Document Frequency (IDF) pénalise les termes présents dans de nombreux documents et favorise ceux qui apparaissent dans moins de documents. Ces derniers peuvent être plus discriminants, sans être nécessairement plus pertinents.

$$\mathrm{idf}(t) = \log\left(\frac{N}{\mathrm{df}(t)}\right)$$

Exemple : corpus de 100 documents.

Terme	**Nombre de documents**	**IDF score**
chat	70	0.36
lait	9	2.41
manger	80	0.22
feu	5	3.00
...	...	...

Pondération TF–IDF

- Le score TF–IDF est le produit de la fréquence locale et de la rareté documentaire.

$$\mathrm{tfidf}(t, d) = \mathrm{tf}(t, d)\,\mathrm{idf}(t)$$

Terme	TF score	IDF score	TF-IDF score
chat	0.015	0.36	0.0054
lait	0.004	2.41	0.0096
manger	0.006	0.22	0.0013
feu	0.003	3.00	0.0090
...	...	...	...

Modélisation thématique

- Décrire chaque document par un mélange latent de thématiques.
- Décrire chaque thématique par une distribution sur le vocabulaire.
- Explorer, organiser ou résumer un corpus sans labels de classes.

Interprétation

Une thématique est une composante statistique, pas nécessairement un concept humain stable. Le nombre de thèmes, les priors, la tokenisation et le corpus modifient fortement le résultat.

LDA : modèle génératif

Pour chaque thème k, tirer une distribution de mots :

$$\phi_k \sim \text{Dirichlet}(\eta)$$

Pour chaque document d, tirer un mélange de thèmes :

$$\theta_d \sim \text{Dirichlet}(\alpha)$$

Puis, pour chaque position n :

$$z_{dn} \sim \text{Categorical}(\theta_d), \qquad w_{dn} \sim \text{Categorical}(\phi_{z_{dn}})$$

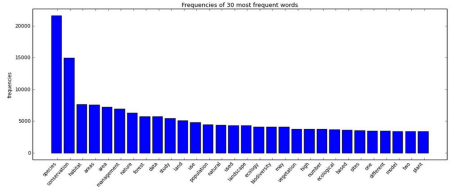
Inférence : estimer la postérieure de θ, ϕ, z par Gibbs, variationnel ou méthodes approchées.

Validation : cohérence, stabilité, diversité, inspection humaine et utilité aval ; la perplexité seule est insuffisante.

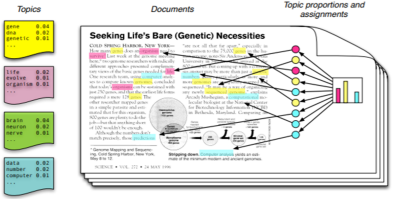
Source : Blei, Ng & Jordan, 2003.

LDA : deux distributions latentes

Thème k : distribution ϕ_k sur les mots



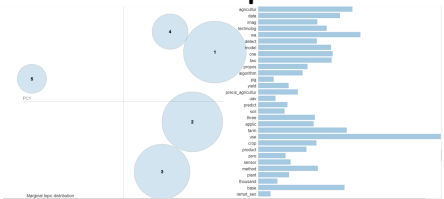
Document d : distribution θ_d sur les thèmes



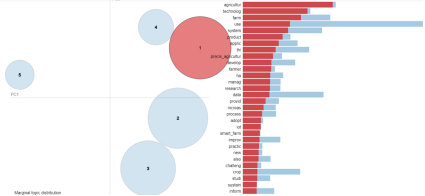
LDA n'assigne pas un thème unique à un document : il estime un mélange, et chaque occurrence de mot possède une variable latente z_{dn}.

LDA : exploration et interprétation d'un corpus

Documents dans l'espace des thèmes



Mots dominants d'un thème



Remarque

Une visualisation ou une liste de mots facilite l'inspection ; elle ne démontre ni la stabilité des thèmes ni leur utilité pour une tâche aval.

Chaîne supervisée : une référence difficile à battre

1. Définir unité de prédiction et coût d'erreur.
2. Construire un découpage sans fuite.
3. Vectoriser : TF–IDF de mots et/ou caractères.
4. Ajuster une régression logistique ou un SVM linéaire.
5. Calibrer le seuil sur validation.

Régression logistique

$$p_\theta(y = 1 \mid x) = \sigma(w^\top x + b)$$

$$\min_\theta - \sum_i \log p_\theta(y_i \mid x_i) + \lambda \|w\|_2^2$$

Les n-grammes de caractères sont robustes aux flexions, fautes et variations orthographiques.

Naive Bayes multinomial : dérivation

Pour un document représenté par les comptes x_j :

$$\hat{y} = \arg\max_c \left[\log P(c) + \sum_{j=1}^{|V|} x_j \log P(w_j \mid c) \right]$$

$$\widehat{P}(w_j \mid c) = \frac{N_{jc} + \alpha}{\sum_k N_{kc} + \alpha |V|}$$

Forces

Très rapide, interprétable, efficace sur données rares et hautement dimensionnelles.

Hypothèse

Indépendance conditionnelle des occurrences sachant la classe : fausse linguistiquement, mais souvent utile statistiquement.

Référence recommandée : comparer NB, régression logistique et SVM avant d'attribuer un gain à l'apprentissage profond.

Étiquetage de séquences : HMM, CRF, décodage

HMM génératif

$$p(x, y) = p(y_1) \prod_{t=2}^{T} p(y_t \mid y_{t-1}) \prod_{t=1}^{T} p(x_t \mid y_t)$$

Les hypothèses portent sur transitions et émissions.

CRF linéaire discriminatif

$$p_\theta(y \mid x) = \frac{\exp s_\theta(x, y)}{\sum_{y'} \exp s_\theta(x, y')}$$

$$s_\theta(x, y) = \sum_t \theta^\top f(y_{t-1}, y_t, x, t)$$

Viterbi calcule $\arg\max_y s(x, y)$ **en** $O(TK^2)$**. Forward–backward calcule la normalisation et les marginales.**

NER : schéma BIO et contraintes structurées

Token	Marie	Curie	travaille	à	Paris	.
Tag	B-PER	I-PER	O	O	B-LOC	O

- BIO/BILOU encode les frontières de spans.
- Une prédiction locale peut créer `I-PER` après `O`.
- Un CRF ou un décodage contraint élimine les transitions invalides.

Évaluation correcte

Une entité est correcte si **type et frontières** sont exacts :

$$F_1^{span} = \frac{2PR}{P+R}$$

L'exactitude par token masque les erreurs de frontières et la classe majoritaire `O`.

Découpage des données : prévenir la fuite

Découpage IID insuffisant si :

- plusieurs textes viennent du même auteur/client ;
- des doublons ou quasi-doublons existent ;
- le déploiement prédit le futur ;
- une source ou un domaine domine le corpus.

Alternatives

- partitionnement par groupe ou entité ;
- partitionnement temporel ;
- test hors domaine ;
- déduplication avant séparation ;
- prétraitements ajustés *uniquement* sur train.

Métriques de classification : micro, macro, pondérée

$$P_c = \frac{TP_c}{TP_c + FP_c}, \qquad R_c = \frac{TP_c}{TP_c + FN_c}, \qquad F_{1,c} = \frac{2P_cR_c}{P_c + R_c}$$

Agrégation	**Calcul**	**Interprétation**
Macro	moyenne des métriques par classe	chaque classe pèse autant ; sensible aux classes rares
Micro	agrégation des TP, FP, FN	toutes les décisions élémentaires sont agrégées ; dominée par les classes fréquentes
Pondérée	moyenne par classe pondérée par support	compromis descriptif, peut masquer une classe critique

Le seuil de décision est un paramètre opérationnel. Choisir une métrique cohérente avec le coût réel des faux positifs et faux négatifs.

Évaluer un modèle de langue et une génération

Modélisation de langue

$$\mathcal{L}_{\mathrm{NLL}} = -\frac{1}{N}\sum_{t=1}^{N} \log p_\theta(x_t \mid x_{<t})$$

$$\mathrm{PPL} = \exp(\mathcal{L}_{\mathrm{NLL}})$$

La perplexité n'est comparable qu'à tokenisation et corpus identiques.

Génération conditionnelle

- BLEU : précision d'n-grammes + pénalité de brièveté.
- ROUGE : recouvrement, souvent rappel.
- BERTScore : similarité d'embeddings contextuels.

Source : Papineni et al., 2002 ; Lin, 2004 ; Zhang et al., 2020.

Incertitude et significativité

Pour comparer deux systèmes sur le même jeu de test :

1. calculer la différence $\Delta = m(A) - m(B)$;
2. rééchantillonner les exemples **par paires** avec remplacement ;
3. recalculer $\Delta^{(b)}$ sur B réplications ;
4. rapporter l'intervalle percentile, par exemple à 95 %.

Pourquoi le bootstrap apparié ?

Il conserve la corrélation entre les prédictions des deux modèles et quantifie l'incertitude due au choix fini des exemples.

- segmenter l'analyse par langue, longueur, source et classe ;
- inspecter qualitativement les désaccords ;
- ne pas transformer une différence non robuste en conclusion générale.

PARTIE III

Représentations distribuées

Introduction aux embeddings

Un embedding est une application apprise

$$E : V \longrightarrow \mathbb{R}^d, \qquad d \ll |V|,$$

qui associe à chaque unité lexicale un vecteur dense.

Ce que l'objectif impose

- proximité distributionnelle ;
- régularités utiles à la tâche ;
- géométrie dépendante du corpus.

Ce qu'il n'impose pas

- une sémantique universelle ;
- l'absence de biais ;
- l'identifiabilité des axes.

Un embedding doit être interprété avec son objectif, ses données et la mesure de similarité utilisée.

Encodage one-hot et embeddings

Vecteur one-hot

Pour $i \in \{1, \dots, |V|\}$, $e_i \in \mathbb{R}^{|V|}$ contient un seul 1.

$$e_i^\top e_j = 0 \quad (i \neq j).$$

- représentation exacte de l'identité ;
- aucune proximité entre unités ;
- dimension $|V|$.

Embedding appris

Une matrice $E \in \mathbb{R}^{|V| \times d}$ transforme

$$v_i = E^\top e_i = E_{i,:}^\top.$$

- dimension $d \ll |V|$;
- paramètres partagés par la tâche ;
- proximités induites par l'apprentissage.

Le produit par une matrice d'embedding équivaut à sélectionner une ligne de E.

Similarité cosinus

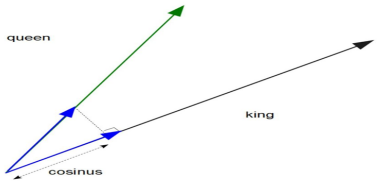
L'hypothèse distributionnelle relie proximité d'usage et proximité vectorielle :

> *"You shall know a word by the company it keeps." — J. R. Firth (1957)*

Pour deux vecteurs non nuls :

$$\cos(x, y) = \frac{x^\top y}{\|x\|\|y\|}.$$

Elle compare les directions et ignore leur norme.



La distance $1 - \cos(x, y)$ n'est pas une métrique sur tout $\mathbb{R}^d$: elle ne satisfait pas toujours l'inégalité triangulaire.

Construction d'une représentation distribuée

Considérons le corpus suivant à titre d'exemple : **cnn in crop analysis**

cnn and svm are widely used.
linear_regression performed along with svm
linear_regression for crop and farm
svm being used for farm monitoring
Do cnn, svm and linear_regression appear in the same context?

Le vocabulaire est alors :

`[cnn, in, crop, analysis, and, svm, are, widely, used, linear_regression, performed, along, with, for, farm, being, monitoring, do, appear, the, same, context]`

$$\dim(\text{vocabulaire}) = 22$$

Matrice de co-occurrence

La matrice de co-occurrence représente la fréquence à laquelle les mots apparaissent ensemble deux à deux.

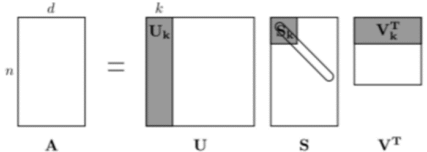
	cnn	in	crop	...
cnn	0	2	1	1	1	2	1	1	1	1	0	0	0	0	0	0	0	1	1	1	1	1
in	2	0	1	1	0	1	0	0	0	1	0	0	0	0	0	0	0	1	1	1	1	1
crop	1	1	0	1	1	0	0	0	0	1	0	0	0	1	1	0	0	0	0	0	0	0
...	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	1	0	0	1	1	1	1	1	0	0	0	1	1	0	0	0	0	0	0	0
	2	1	0	0	1	0	1	1	2	2	1	1	1	1	1	1	1	1	1	1	1	1
	1	0	0	0	1	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	0	0	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	0	0	1	2	1	1	0	0	0	0	0	1	1	1	1	0	0	0	0	0
	1	1	1	0	1	2	0	0	0	0	1	1	1	1	1	0	0	1	1	1	1	1
	0	0	0	0	0	1	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	1	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	1	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0
	0	0	1	0	1	1	0	0	1	1	0	0	0	0	2	1	1	0	0	0	0	0
	0	0	1	0	1	1	0	0	1	1	0	0	0	2	0	1	1	0	0	0	0	0
	0	0	0	0	0	1	0	0	1	0	0	0	0	1	1	0	1	0	0	0	0	0
	0	0	0	0	0	1	0	0	1	0	0	0	0	1	1	1	0	0	0	0	0	0
	1	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	1	1	1	1
	1	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	1	0	1	1	1
	1	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	1	1	0	1	1
	1	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	1	1	1	0	1
	1	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	1	1	1	1	0

Réduction de la dimension

La décomposition en valeurs singulières est une technique permettant de réduire la dimension des données (similaire à l'ACP).

Étant donné une matrice $A \in \mathbb{R}^{n \times d}$

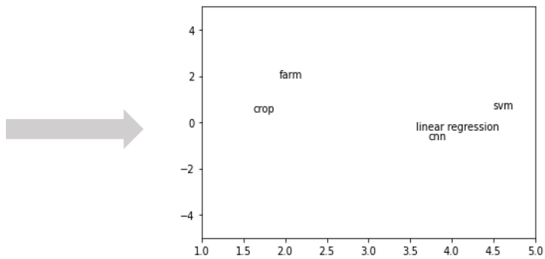
$$A = UDV^T \quad \text{où} \quad U \in \mathbb{R}^{n \times r}, D \in \mathbb{R}^{r \times r}, V \in \mathbb{R}^{d \times r}.$$



Une représentation de rang r est généralement construite à partir de $U_r D_r$ (ou $U_r D_r^{1/2}$ selon la convention), et non de U_r seul.

Visualisation des représentations distribuées

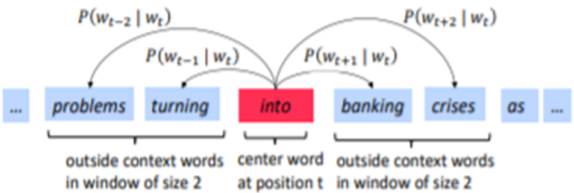
```
cnn  [ 3.72113518, -0.73233585],
In   [ 2.7940387 , -1.40864784],
crop [ 1.61178082,  0.44786021],
...  [ 0.77784994, -0.31230389],
     [ 2.12245048,  1.29571556],
     [ 4.49293777,  0.58090417],
     [ 1.33312748,  0.66758743],
     [ 1.33312748,  0.66758743],
     [ 2.25882809,  1.80738544],
     [ 3.56593605, -0.31006976],
     [ 0.95394179,  0.079159  ],
     [ 0.95394179,  0.079159  ],
     [ 0.95394179,  0.079159  ],
     [ 1.92921553,  1.94783497],
     [ 1.92921553,  1.94783497],
     [ 1.12301312,  1.42126096],
     [ 1.12301312,  1.42126096],
     [ 2.26025282, -1.31571175],
     [ 2.26025282, -1.31571175],
     [ 2.26025282, -1.31571175],
     [ 2.26025282, -1.31571175],
     [ 2.26025282, -1.31571175]]
```



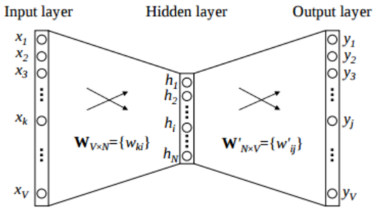
Une projection en deux dimensions déforme nécessairement certaines distances : elle sert à explorer, non à prouver la séparation des groupes.

Word2Vec : principes

- **Principes de base :**
  - Word2Vec exploite l'hypothèse distributionnelle : des contextes similaires conduisent souvent à des usages syntaxiques ou sémantiques proches.
  - Utilise un réseau de neurones peu profond pour apprendre des embeddings de mots à partir de grands corpus.
- **Deux architectures principales :**
  1. *Continuous Bag of Words (CBOW)* : Prédit le mot cible à partir du contexte.
  2. *Skip-Gram* : Prédit le contexte à partir du mot cible.



Modèle CBOW : formulation mathématique



Le contexte est agrégé sans tenir compte de son ordre.

Pour prédire le mot cible w_t à partir du contexte C :

$$h = \frac{1}{|C|} \sum_{c \in C} v_c,$$

$$P(w_t \mid C) = \frac{\exp(u_{w_t}^\top h)}{\sum_{w \in V} \exp(u_w^\top h)}.$$

- v_c : embedding d'entrée ;
- u_w : embedding de sortie ;
- perte exacte : $-\log P(w_t \mid C)$.

Le softmax exact coûte $O(|V|d)$ par exemple. Le softmax hiérarchique le factorise ; l'échantillonnage négatif le remplace par un objectif logistique binaire.

Modèle Skip-Gram : formulation mathématique

- **Objectif :** Prédire les mots de contexte à partir du mot cible w_t.
- **Fonction de prédiction :**

$$P(C|w_t) = \prod_{c \in C} \frac{\exp(u_c^\top v_{w_t})}{\sum_{w \in V} \exp(u_w^\top v_{w_t})} \tag{1}$$

- **Où :**
  - v_{w_t} est l'embedding d'entrée du mot central ;
  - u_c est l'embedding de sortie d'un mot de contexte.
- **Optimisation :** Minimisation de la fonction de coût, souvent une forme de cross-entropy ou log-likelihood négatif.

Word2Vec : calcul efficace

Le coût du softmax complet motive des objectifs et des schémas d'échantillonnage plus efficaces.

Échantillonnage négatif

- remplace la vraisemblance softmax par des décisions logistiques binaires ;
- met à jour le positif et k négatifs au lieu de tout le vocabulaire ;
- apprend des représentations adaptées à la discrimination observé/non observé.

Sous-échantillonnage

- écarte aléatoirement une partie des occurrences très fréquentes ;
- réduit leur domination dans les fenêtres ;
- accélère l'entraînement, mais modifie la distribution effective du corpus.

Échantillonnage négatif : formulation mathématique

Pour une paire observée (i, o) et k contextes négatifs n_j :

$$\mathcal{L} = -\log \sigma(u_o^\top v_i) - \sum_{j=1}^{k} \log \sigma(-u_{n_j}^\top v_i), \qquad n_j \sim P_n$$

Terme positif

$$-\log \sigma(u_o^\top v_i)$$

rapproche le mot central et son contexte observé.

Termes négatifs

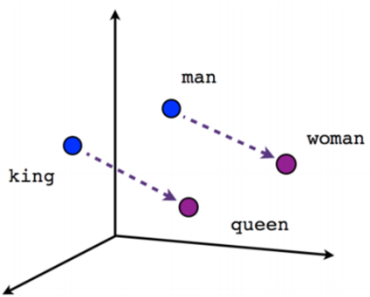
$$-\sum_j \log \sigma(-u_{n_j}^\top v_i)$$

séparent les paires tirées dans P_n.

En pratique, $P_n(w) \propto U(w)^{3/4}$, où $U(w)$ est la fréquence brute. Le choix de k règle le compromis coût–signal d'apprentissage.

Source : Mikolov et al., 2013.

Applications pratiques de Word2Vec



- recherche de voisins et expansion lexicale ;
- initialisation de modèles supervisés ;
- regroupement exploratoire de vocabulaires ;
- analogies comme test de régularités géométriques.

Les analogies vectorielles sont des régularités empiriques du corpus ; elles ne constituent ni une règle algébrique générale ni une garantie de neutralité.

Autres représentations distribuées

Il existe d'autres représentations distribuées :

- GloVe (2014) : apprend des vecteurs par régression pondérée sur le logarithme des comptes de cooccurrence, avec un objectif motivé par les rapports de probabilités.
- fastText (à partir de 2016) : le modèle représente un mot à partir de ses (n)-grammes de caractères. Il peut ainsi construire un vecteur pour un mot absent du vocabulaire, dès lors que ses sous-unités sont connues.
- Représentations contextuelles (Transformers), etc.

PARTIE IV

Réseaux récurrents et modèles de séquences

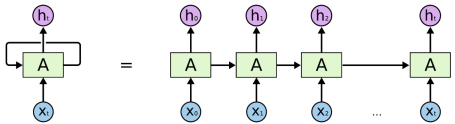
RNN : repères historiques

Période	Évolution
1986–1990	rétropropagation temporelle et réseaux d'Elman/Jordan ;
1991	analyse du gradient sur de longues dépendances ;
1997	LSTM : chemin mémoire additif et portes ;
2014	GRU et essor des encodeurs–décodeurs récurrents ;
2017	les Transformers remplacent la récurrence par l'attention dans de nombreuses tâches de TAL.

L'enjeu central n'est pas seulement de mémoriser le passé, mais de propager un signal d'apprentissage sur de nombreux pas de temps.

Source : Rumelhart et al., 1986 ; Hochreiter, 1991 ; Hochreiter & Schmidhuber, 1997.

RNN : état récurrent et paramètres partagés



- la cellule reçoit x_t et l'état précédent h_{t-1} ;
- les mêmes paramètres sont réutilisés à chaque position ;
- h_t résume le préfixe $x_{1:t}$, sous contrainte de dimension fixe ;
- le graphe déroulé rend visibles les chemins suivis par le gradient.

La récurrence impose une dépendance séquentielle : les états ne peuvent pas tous être calculés en parallèle.

RNN simple : formulation et dimensions

Pour $x_t \in \mathbb{R}^{d_x}$, $h_t \in \mathbb{R}^{d_h}$ et $y_t \in \mathbb{R}^{d_y}$:

$$a_t = W_{xh}x_t + W_{hh}h_{t-1} + b_h, \qquad h_t = \phi(a_t),$$

$$o_t = W_{hy}h_t + b_y, \qquad p(y_t \mid x_{1:t}) = \mathrm{softmax}(o_t).$$

Paramètre	**Dimension**	**Rôle**
W_{xh}	$d_h \times d_x$	projection de l'entrée ;
W_{hh}	$d_h \times d_h$	dynamique récurrente ;
W_{hy}	$d_y \times d_h$	projection vers la sortie.

La capacité mémoire dépend de la dynamique induite par W_{hh} et ϕ, pas d'une mémoire symbolique explicite.

BPTT : dérivation du gradient

Soit $L = \sum_{u=1}^{T} \ell_u$. Un paramètre partagé θ agit à chaque pas de temps :

$$\frac{\partial L}{\partial \theta} = \sum_{u=1}^{T} \sum_{t=1}^{u} \frac{\partial \ell_u}{\partial h_u} \left(\prod_{k=t+1}^{u} J_k \right) \frac{\partial h_t}{\partial \theta}, \qquad J_k = \frac{\partial h_k}{\partial h_{k-1}}.$$

Le même calcul s'écrit récursivement avec l'adjoint $\delta_t = \partial L / \partial h_t$:

$$\delta_t = \frac{\partial \ell_t}{\partial h_t} + J_{t+1}^{\top} \delta_{t+1}, \qquad \frac{\partial L}{\partial \theta} = \sum_{t=1}^{T} \delta_t^{\top} \frac{\partial h_t}{\partial \theta}.$$

BPTT est la rétropropagation ordinaire appliquée au graphe du RNN déroulé dans le temps.

BPTT : disparition, explosion et troncature

Pour un RNN simple,

$$J_k = \operatorname{diag}(\phi'(a_k))\, W_{hh}, \qquad \left\| \frac{\partial h_u}{\partial h_t} \right\| \leq \prod_{k=t+1}^{u} \|J_k\|.$$

Conséquences

- normes durablement inférieures à 1 : disparition ;
- normes supérieures à 1 : explosion ;
- le conditionnement dépend aussi de l'orientation des Jacobiennes.

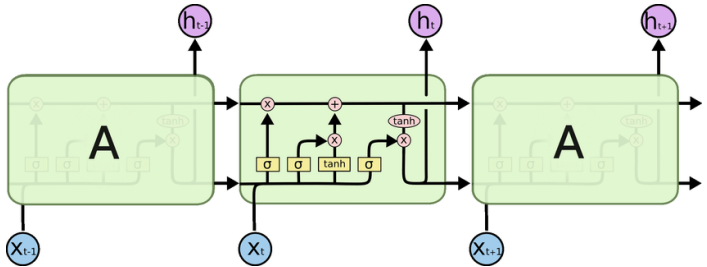
Réponses usuelles

- chemins additifs des LSTM/GRU ;
- initialisation et normalisation ;
- écrêtage pour l'explosion ;
- BPTT tronquée : coût réduit, dépendances longues biaisées.

RNN avec Long Short-Term Memory (LSTM)

Les LSTM introduisent un état de cellule c_t dont la mise à jour est principalement additive :

- f_t sélectionne la mémoire conservée ;
- i_t pondère l'information candidate ;
- o_t contrôle l'exposition de la cellule dans h_t.



Le chemin additif de c_t facilite la propagation du gradient, sans garantir à lui seul l'apprentissage de dépendances arbitrairement longues.

Formulation mathématique des LSTM

Avec $z_t = [h_{t-1}; x_t]$:

$$\begin{aligned} f_t &= \sigma(W_f z_t + b_f), & i_t &= \sigma(W_i z_t + b_i), \\ \tilde{c}_t &= \tanh(W_c z_t + b_c), & o_t &= \sigma(W_o z_t + b_o), \\ c_t &= f_t \odot c_{t-1} + i_t \odot \tilde{c}_t, & h_t &= o_t \odot \tanh(c_t). \end{aligned}$$

- les portes sont des vecteurs dans $[0,1]^{d_h}$;
- $\odot$ agit coordonnée par coordonnée ;
- c_t porte la mémoire, h_t l'état exposé à la couche suivante.

Gradient le long de la cellule

Sur le chemin additif direct, en maintenant portes et candidat fixes, $\partial c_t / \partial c_{t-1} = \mathrm{diag}(f_t)$: la porte d'oubli règle directement la conservation du signal sur ce chemin.

RNN avec Gated Recurrent Unit (GRU)

Le GRU fusionne état de cellule et état caché. Dans la convention suivante :

$$
\begin{aligned}
z_t &= \sigma(W_z[h_{t-1}; x_t] + b_z), \\
r_t &= \sigma(W_r[h_{t-1}; x_t] + b_r), \\
\tilde{h}_t &= \tanh(W_h[r_t \odot h_{t-1}; x_t] + b_h), \\
h_t &= (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t.
\end{aligned}
$$

Par rapport au LSTM

- deux portes au lieu de trois ;
- moins de paramètres à largeur égale.

À comparer empiriquement

- qualité à budget identique ;
- vitesse, mémoire et stabilité.

PARTIE IV – SUITE

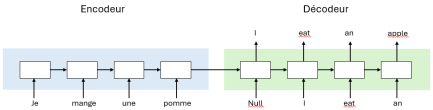
Seq2Seq et attention neuronale

Seq2Seq : factorisation et architecture

Pour une entrée $x_{1:T_x}$ et une sortie $y_{1:T_y}$ de longueurs variables :

$$p_\theta(y_{1:T_y} \mid x_{1:T_x}) = \prod_{t=1}^{T_y} p_\theta(y_t \mid y_{<t}, x_{1:T_x}).$$

- l'encodeur construit une représentation de la source ;
- le décodeur génère la cible de façon autorégressive ;
- le cadre couvre traduction, résumé et transformation conditionnelle.



Seq2Seq désigne une factorisation entrée–sortie ; elle n'impose ni RNN ni Transformer.

Seq2Seq récurrent : encodeur et décodeur

Encodeur

$$h_j = f_{\text{enc}}(x_j, h_{j-1}), \qquad c = h_{T_x}.$$

Le dernier état c doit résumer toute la source dans l'architecture historique sans attention.

Décodeur

$$s_t = f_{\text{dec}}(y_{t-1}, s_{t-1}, c),$$

$$p_t = \text{softmax}(W_o s_t).$$

La génération s'arrête au symbole de fin ou à une longueur maximale.

Remarque

Le vecteur fixe c crée un goulot d'étranglement : l'attention le remplacera par un contexte c_t dépendant du pas de décodage.

Seq2Seq récurrent : apports et limites

Apports

- sorties de longueur variable ;
- apprentissage conditionnel de bout en bout ;
- même cadre pour traduction, résumé, parole et transformations structurées.

Limites

- compression de la source dans un état fixe ;
- calcul récurrent peu parallélisable ;
- erreurs cumulatives au décodage ;
- dépendances longues difficiles à apprendre.

L'attention traite d'abord le goulot d'étranglement de la source ; le Transformer supprimera ensuite la récurrence.

Entraînement seq-to-seq : teacher forcing

$$\mathcal{L}(\theta) = -\sum_{t=1}^{T_y} \log p_\theta(y_t \mid y_{<t}, x)$$

- **En entraînement :** le décodeur reçoit généralement le vrai préfixe $y_{<t}$, ce qui permet de calculer une perte à chaque position et facilite l'optimisation. Un décodeur récurrent reste séquentiel ; un Transformer causal traite les positions en parallèle pendant l'entraînement.
- **En inférence :** il reçoit ses propres tokens générés ; une erreur modifie tous les préfixes suivants.
- Cette divergence est appelée *exposure bias*. Scheduled sampling, objectifs de séquence et distillation proposent des compromis, sans solution universelle.

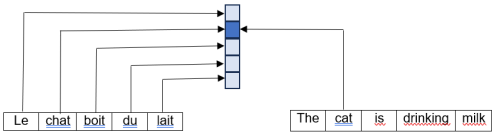
Remarque

La log-vraisemblance token par token n'optimise pas directement BLEU, factualité ou utilité de la séquence complète.

Attention encodeur–décodeur : motivation

Au lieu d'imposer un unique vecteur source c, le décodeur construit un contexte c_t différent à chaque étape.

- tous les états de l'encodeur restent accessibles ;
- le modèle apprend un alignement souple entre source et cible ;
- le chemin d'information entre deux positions est raccourci.



Attention additive : score, poids et contexte

Pour l'état du décodeur s_{t-1} et les états de l'encodeur h_j :

$$e_{tj} = v_a^\top \tanh(W_s s_{t-1} + W_h h_j), \qquad \alpha_{tj} = \frac{\exp(e_{tj})}{\sum_{k=1}^{T_x} \exp(e_{tk})},$$

$$c_t = \sum_{j=1}^{T_x} \alpha_{tj} h_j, \qquad \sum_j \alpha_{tj} = 1.$$

Interprétation calculatoire

- e_{tj} mesure une compatibilité ;
- α_{tj} pondère la source pour le pas t.

Coût

- $O(T_x T_y d)$ pour l'alignement dense ;
- les états source sont réutilisés à chaque pas.

Source : Bahdanau, Cho & Bengio, 2015.

Attention : principales formes

Forme	Requêtes	Clés et valeurs	Usage
Encodeur–décodeur	états du décodeur	états de la source	génération conditionnelle ;
Self-attention	positions de la séquence	même séquence	contextualisation ;
Locale ou creuse	positions courantes	sous-ensemble autorisé	coût réduit, biais de localité.

Fonction de score

L'attention additive et le produit scalaire sont deux paramétrisations de la compatibilité entre requêtes et clés. Elles ne changent pas, à elles seules, l'ensemble des positions accessibles.

Attention : apports et limites

Apports

- contexte conditionné par la requête ;
- alignement appris de bout en bout ;
- chemins courts entre positions éloignées ;
- parallélisation de la self-attention.

Limites

- coût quadratique pour l'attention dense ;
- pondération interne $\neq$ explication causale ;
- dépendances autorisées fixées par le masque ;
- l'attention n'assure ni factualité ni raisonnement.

Le Transformer conserve l'attention, supprime la récurrence et ajoute résidus, normalisation, réseaux feed-forward et information de position.

Source : Jain & Wallace, 2019.

PARTIE V

Transformers

Architecture du Transformer originel

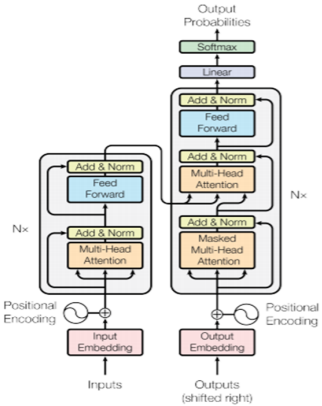


Figure 1: The Transformer - model architecture.

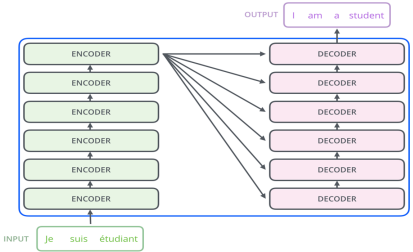
Source : Vaswani et al., 2017.

- **Encodeur :** self-attention bidirectionnelle et réseau feed-forward.
- **Décodeur :** self-attention causale, cross-attention vers l'encodeur et réseau feed-forward.
- **Chaque sous-couche :** connexion résiduelle et normalisation.

L'architecture sépare la représentation de la source et la génération conditionnelle de la cible.

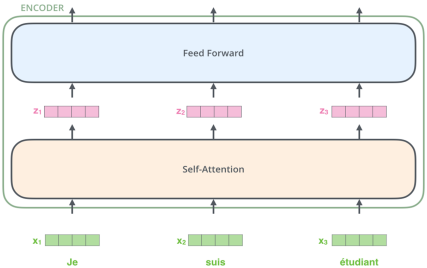
Cross-attention et self-attention

Cross-attention



Requêtes du décodeur ; clés et valeurs de l'encodeur.

Self-attention

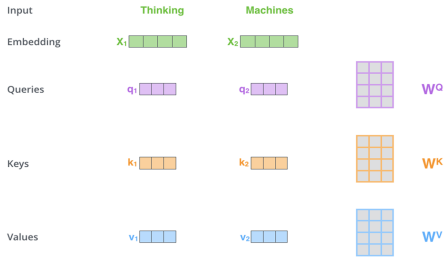


Requêtes, clés et valeurs issues de la même séquence.

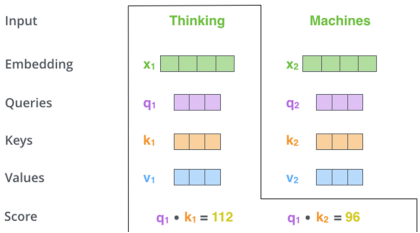
La provenance de Q, K et V distingue ces deux opérations ; la formule d'attention reste la même.

Self-attention 1/2 : projections et compatibilités

1. Construire Q, K, V



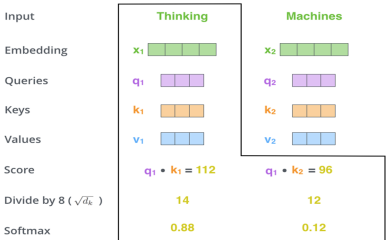
2. Calculer $QK^\top$



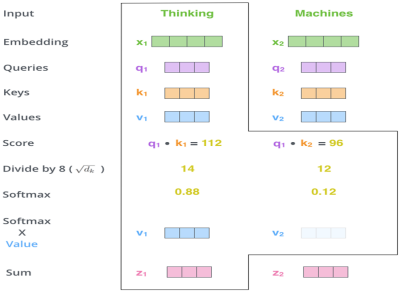
Chaque score compare une requête à une clé ; les valeurs n'interviennent qu'après normalisation.

Self-attention 2/2 : pondération et agrégation

3. Normaliser les scores

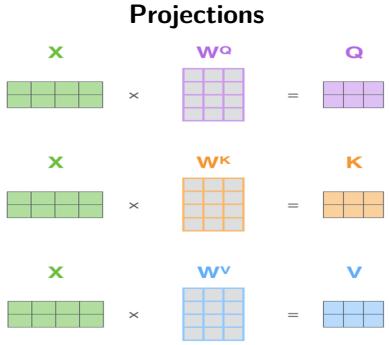


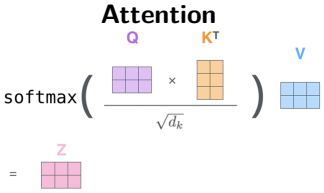
4. Combiner les valeurs



Le softmax produit une distribution par requête ; la sortie est une combinaison convexe des vecteurs valeur correspondants.

Self-attention : écriture matricielle

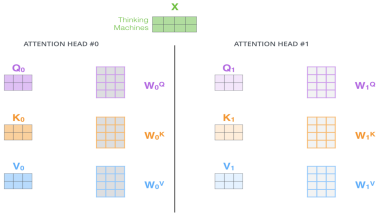




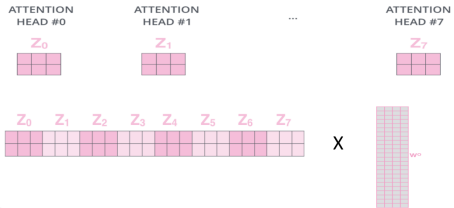
$$Q = XW^Q, \quad K = XW^K, \quad V = XW^V, \qquad \mathrm{Att}(Q,K,V) = \mathrm{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V.$$

Multi-head attention : calcul et recombinaison

Projections propres à chaque tête



Concaténation puis projection



$$\mathrm{MHA}(X) = \mathrm{Concat}(H_1, \ldots, H_H) W^O.$$

Performances du Transformer originel

Le modèle de Vaswani et al. est évalué sur WMT 2014 anglais–allemand (environ 4,5 millions de paires de phrases) et anglais–français (environ 36 millions de paires).

- 8 GPU NVIDIA P100
- **Base** : 12 heures
- **Large** : 3,5 jours

Table 2: The Transformer achieves better BLEU scores than previous state-of-the-art models on the English-to-German and English-to-French newstest2014 tests at a fraction of the training cost.

Model	BLEU		Training Cost (FLOPs)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [18]	23.75			
Deep-Att + PosUnk [39]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [38]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [9]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [32]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
Deep-Att + PosUnk Ensemble [39]		40.4		$8.0 \cdot 10^{20}$
GNMT + RL Ensemble [38]	26.30	41.16	$1.8 \cdot 10^{20}$	$1.1 \cdot 10^{21}$
ConvS2S Ensemble [9]	26.36	**41.29**	$7.7 \cdot 10^{19}$	$1.2 \cdot 10^{21}$
Transformer (base model)	27.3	38.1	**$3.3 \cdot 10^{18}$**	
Transformer (big)	**28.4**	**41.8**	$2.3 \cdot 10^{19}$	

Source : Vaswani et al., *Attention Is All You Need*, 2017.

Self-attention : dimensions et projection

Soit $X \in \mathbb{R}^{n \times d_{\text{model}}}$. Pour une tête :

$$Q = XW_Q, \quad K = XW_K, \quad V = XW_V,$$

avec $W_Q, W_K \in \mathbb{R}^{d_{\text{model}} \times d_k}$ et $W_V \in \mathbb{R}^{d_{\text{model}} \times d_v}$.

$$\text{Att}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}} + M\right) V$$

Pourquoi $1/\sqrt{d_k}$?

Si les composantes sont centrées et de variance 1, le produit scalaire a une variance proportionnelle à d_k. Le facteur évite la saturation du softmax.

Rôle du masque M

$M_{ij} = 0$ si j est visible ; $M_{ij} = -\infty$ sinon. Il encode padding, causalité ou contraintes structurées.

Multi-head attention et bloc Transformer

$$\mathrm{MHA}(X) = \mathrm{Concat}(H_1, \dots, H_h) W_O, \qquad H_i = \mathrm{Att}(XW_Q^i, XW_K^i, XW_V^i)$$

Bloc pré-norm moderne

$$X' = X + \mathrm{MHA}(\mathrm{LN}(X))$$
$$Y = X' + \mathrm{FFN}(\mathrm{LN}(X'))$$
$$\mathrm{FFN}(x) = W_2 \phi(W_1 x + b_1) + b_2$$

- Les résidus facilitent la propagation du gradient.
- La normalisation stabilise l'échelle des activations.
- Le FFN agit indépendamment à chaque position.
- Plusieurs têtes n'impliquent pas automatiquement plusieurs phénomènes linguistiques interprétables.

Masque causal : factorisation autorégressive

$$p_\theta(x_{1:T}) = \prod_{t=1}^{T} p_\theta(x_t \mid x_{<t})$$

La ligne t de la matrice d'attention ne peut consulter que les positions $j \leq t$.

$$M = \begin{bmatrix} 0 & -\infty & -\infty & -\infty \\ 0 & 0 & -\infty & -\infty \\ 0 & 0 & 0 & -\infty \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Remarque

L'entraînement peut traiter en parallèle toutes les positions connues grâce au masque ; la génération reste séquentielle car le token x_t n'existe pas avant d'avoir généré $x_{<t}$.

Information de position : absolue, relative, RoPE

Sinusoïdes absolues

$$PE_{p,2i} = \sin\left(p/10000^{2i/d}\right)$$

$$PE_{p,2i+1} = \cos\left(p/10000^{2i/d}\right)$$

Elles sont ajoutées aux embeddings.

RoPE

Une rotation dépendant de la position est appliquée aux requêtes et clés :

$$q'_p = R_p q_p, \qquad k'_s = R_s k_s$$

Ainsi $q'^{\top}_p k'_s$ dépend de la position relative $s - p$.

L'extrapolation à des contextes plus longs que ceux vus à l'entraînement n'est pas garantie ; elle dépend du schéma positionnel et de la méthode d'extension.

Complexité : où part le calcul ?

Opération	Complexité	Conséquence
Projections Q, K, V, O	$O(nd^2)$	dominante lorsque $d \gg n$
Scores $QK^\top$	$O(n^2 d)$	mémoire et calcul quadratiques en longueur
Application à V	$O(n^2 d)$	même dépendance quadratique
FFN	$O(n d d_{\text{ff}})$	souvent grande part des FLOP et paramètres

- Fenêtrage, attention creuse ou linéaire modifient le compromis exactitude–coût.
- FlashAttention est un algorithme exact *IO-aware* : il réduit les accès mémoire sans changer la définition de l'attention.
- Une complexité asymptotique ne suffit pas : profilage matériel, lot, précision et longueur comptent.

Source : Dao et al., 2022.

Inférence autorégressive et cache KV

À l'étape t, les clés et valeurs des tokens passés sont inchangées. On mémorise :

$$K_{1:t-1}, V_{1:t-1}$$

et on ne calcule que q_t, k_t, v_t.

Compromis

- moins de calcul redondant ;
- mémoire proportionnelle au nombre de couches, à la longueur et au nombre de têtes KV ;
- MQA/GQA partagent des têtes de clés/valeurs pour réduire le cache.

Deux régimes

Prefill : traitement parallèle du prompt, plutôt compute-bound. **Decode** : un token à la fois, souvent memory-bandwidth-bound.

Encodeur, décodeur, encodeur–décodeur

	Attention	**Objectif typique**	**Tâches**
Encodeur	bidirectionnelle	MLM / représentation	classification, NER, recherche
Décodeur	causale	prédiction du token suivant	génération, agents, code
Encodeur–décodeur	encodeur bidirectionnel + décodeur causal + cross-attention	débruitage / seq2seq	traduction, résumé, transformation

Le choix d'architecture impose un biais inductif. Un décodeur peut réaliser de nombreuses tâches via prompting, mais n'est pas toujours le meilleur compromis qualité–latence–coût.

Source : Vaswani et al., 2017 ; Devlin et al., 2019 ; Raffel et al., 2020.

PARTIE VI

Préentraînement et grands modèles de langage

BERT : encodeur pré-entraîné

BERT (*Bidirectional Encoder Representations from Transformers*) est un Transformer encodeur pré-entraîné sur du texte non annoté, puis adapté aux tâches aval.

- **Contexte bidirectionnel :** chaque position non masquée peut consulter la gauche et la droite.
- **Représentations contextuelles :** le vecteur d'un même token dépend de la phrase.
- **Adaptation :** une tête légère est ajoutée, puis le modèle complet ou une partie de ses paramètres est ajustée.
- **Limite :** BERT n'est pas, par conception, un générateur autorégressif de texte long.

Source : Devlin et al., 2019.

Pré-entraînement de BERT

Masked Language Modeling (MLM)
15% des positions sont sélectionnées : 80% reçoivent `[MASK]`, 10% un token aléatoire et 10% restent inchangées.

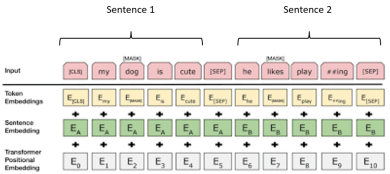
$$L_{\text{MLM}} = -\sum_{t \in \mathcal{M}} \log p_\theta(x_t \mid \widetilde{x}).$$

La perte porte uniquement sur $\mathcal{M}$, les positions sélectionnées dans la séquence corrompue $\widetilde{x}$.

Next Sentence Prediction (NSP)
Le classifieur distingue une phrase B réellement consécutive d'une phrase tirée aléatoirement, avec un échantillonnage 50/50.

$$L_{\text{NSP}} = -y \log \hat{y} - (1-y) \log(1-\hat{y}).$$

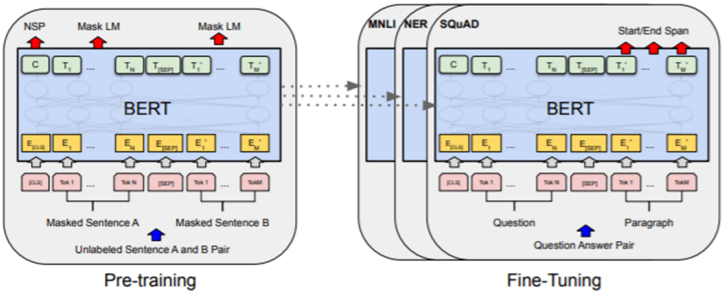
L'objectif original minimise $L_{\text{MLM}} + L_{\text{NSP}}$.



Fine-tuning de BERT pour des tâches spécifiques

Après pré-entraînement, BERT est affiné pour des tâches spécifiques:

- Ajout d'une couche de sortie spécifique à la tâche (classification, NER, QA).
- Fine-tuning de tous les paramètres du modèle pré-entraîné sur le corpus de la tâche.



Performances de BERT

BERT a été pré-entraîné sur BookCorpus (800 millions de mots) et Wikipédia en anglais (2 500 millions de mots).

Deux modèles :

- **BERT Base** : 12 couches avec 110 millions de paramètres.
- **BERT Large** : 24 couches avec 340 millions de paramètres.

System	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k	Average -
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

Modèles autorégressifs : l'exemple GPT-3

GPT-3 est un Transformer décodeur de 175 milliards de paramètres, entraîné à prédire le token suivant.

- **Objectif :** maximiser $\sum_t \log p_\theta(x_t \mid x_{<t})$.
- **In-context learning :** l'instruction et quelques démonstrations conditionnent la génération sans mise à jour des poids.
- **Résultat empirique :** les performances few-shot augmentent avec l'échelle sur plusieurs tâches, mais restent hétérogènes et sensibles au prompt.
- **Limites :** factualité non garantie, biais du corpus, coût et fenêtre de contexte finie.

Source : Brown et al., 2020.

Architecture et objectif de GPT-3

GPT-3 utilise une pile de blocs Transformer à attention causale.

- **Taille :** 96 couches, dimension cachée 12 288, 96 têtes dans la version 175B.
- **Contexte :** 2 048 tokens dans le modèle publié en 2020.
- **Attention :** motif dense et local alterné dans l'implémentation décrite.
- **Formulation :** si h_{t-1} est l'état de la dernière position du préfixe,

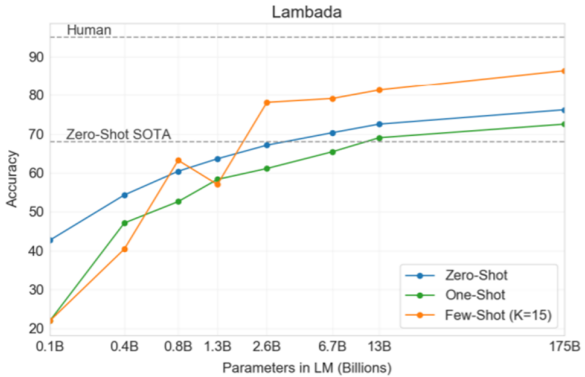
$$p_\theta(w_t \mid w_{<t}) = \mathrm{softmax}(W_o h_{t-1} + b_o)_{w_t}. \tag{2}$$

Apprentissage en contexte avec GPT-3

Le few-shot présenté dans l'article GPT-3 est un apprentissage *en contexte* :

- **Données :** quelques démonstrations (x_i, y_i) sont concaténées dans le prompt.
- **Mécanisme dans GPT-3 :**
  - le modèle prédit la suite conditionnellement aux exemples ;
  - aucun gradient n'est calculé et θ reste inchangé.
- **Sensibilité :** ordre, formulation, format des labels et choix des démonstrations peuvent modifier fortement le résultat.
- **À distinguer :** le fine-tuning modifie les poids ; le prompting ne fait que conditionner l'entrée.

Performances few-shot de GPT-3



Le gain avec l'échelle est net sur cet exemple, mais il n'est ni uniforme entre les tâches ni une preuve d'apprentissage des poids pendant le prompt.

Source : Brown et al., 2020.

ChatGPT : produit, modèles et post-entraînement

ChatGPT est un produit conversationnel, pas une architecture unique. Les modèles sous-jacents et les procédures exactes ont évolué depuis son lancement. Le schéma historique documenté pour

InstructGPT aide à comprendre le post-entraînement :

1. pré-entraînement autorégressif d'un modèle de base ;
2. fine-tuning supervisé sur des démonstrations ;
3. collecte de comparaisons humaines et apprentissage d'un modèle de récompense ;
4. optimisation de la politique sous contrainte de divergence à une référence.

Ne pas inférer d'une procédure publiée pour un modèle historique la recette exacte d'un produit actuel.

Source : Ouyang et al., 2022.

Objectifs de pré-entraînement : quel signal ?

Objectif	Perte	Biais inductif
Causal LM	$-\sum_t \log p(x_t \mid x_{<t})$	génération gauche–droite
MLM	$-\sum_{t\in M} \log p(x_t \mid x_{\setminus M})$	représentation bidirectionnelle
Débruitage seq2seq	$-\log p(x \mid \widetilde{x})$	reconstruction et transformation de séquences
Contrastif	$-\log \frac{e^{s(q,d^+)/\tau}}{\sum_j e^{s(q,d_j)/\tau}}$	espace utile pour recherche / similarité

Un « modèle de fondation » n'est pas universel : objectif, données, architecture et protocole d'adaptation déterminent ses compétences.

Décodage : de la distribution à la séquence

- **Glouton** : $\arg\max_v p(v \mid x_{<t})$.
- **Beam search** : conserve B préfixes ; utile pour sorties contraintes, peut réduire la diversité.
- **Température** :

$$p_T(v) = \mathrm{softmax}(z_v / T)$$

- **Top-k** : échantillonner parmi les k tokens les plus probables.
- **Top-p** : plus petit ensemble dont la masse cumulée dépasse p.
- **Contraintes** : grammaires, JSON Schema, tokens interdits.

Point de méthode

Une génération plus aléatoire n'est pas « plus créative » au sens mesuré. Régler sur une évaluation de tâche, avec seed et paramètres rapportés.

Fine-tuning complet, LoRA et QLoRA

LoRA gèle $W_0 \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ et apprend une mise à jour de faible rang :

$$W = W_0 + \Delta W, \qquad \Delta W = \frac{\alpha}{r} BA$$

$$A \in \mathbb{R}^{r \times d_{\text{in}}}, \quad B \in \mathbb{R}^{d_{\text{out}} \times r}$$

Paramètres entraînés : $r(d_{\text{in}} + d_{\text{out}})$.

- **LoRA** : moins de mémoire optimiseur, adaptateurs modulaires.
- **QLoRA** : poids de base quantifiés pendant l'entraînement des adaptateurs.
- Le rang, les modules ciblés, la qualité des données et le taux d'apprentissage restent déterminants.

Source : Hu et al., 2022 ; Dettmers et al., 2023.

Choisir une stratégie d'adaptation

	Connaissance	**Comportement**	**Quand l'utiliser**
Prompting	contexte fourni	consignes ponctuelles	prototype, tâche déjà maîtrisée
SFT / PEFT	implicitement dans les poids	format, style, procédure	comportement répétitif, exemples nombreux
Pré-entraînement continu	domaine massif	représentation du domaine	jargon/distribution très éloignés, budget élevé

Le fine-tuning vise d'abord un comportement ou une représentation. Il ne garantit ni mémorisation fidèle, ni actualisation, ni traçabilité des faits.

PARTIE VII

Alignement des modèles de langage

Fine-tuning supervisé (SFT)

Le fine-tuning supervisé adapte un modèle de base à une distribution d'instructions, de formats et de comportements cibles. **Méthodologie:**

1. Collecte d'un ensemble de données de dialogues de haute qualité, comprenant des échanges humains et des interactions annotées pour capturer divers contextes conversationnels.
2. Transformation de ces dialogues en paires de prompts et réponses pour créer des exemples d'entraînement qui guident le modèle dans l'apprentissage de structures conversationnelles et de réponses appropriées.
3. Optimisation de l'entropie croisée token par token sur les réponses de démonstration, chaque token étant conditionné par le prompt et le préfixe de réponse attendu.

Alignement des LLM avec RLHF

Objectif régularisé

$$\max_\theta \ \mathbb{E}_{x,y\sim\pi_\theta}[r_\phi(x,y)] - \beta\,\mathbb{E}_x D_{\mathrm{KL}}(\pi_\theta(\cdot|x)\|\pi_{\mathrm{ref}}(\cdot|x))$$

- r_ϕ approxime des préférences annotées.
- La référence limite la dérive loin du SFT.
- L'optimisation porte sur des séquences de tokens.
- « Aligné » signifie aligné avec un protocole et une distribution de préférences donnés.

Source : Ouyang et al., 2022.

Approche RLHF pour l'entraînement des LLM

1. **SFT :** apprendre sur des démonstrations humaines de haute qualité.
2. **Échantillonnage :** générer plusieurs réponses par prompt avec la politique courante.
3. **Préférences :** faire classer les réponses selon une grille explicite ; contrôler accord et qualité.
4. **Modèle de récompense :** apprendre un score à partir des comparaisons.
5. **Policy optimization :** maximiser ce score sous contrainte KL, par exemple avec PPO.

Boucle de Goodhart

Si le modèle exploite une faiblesse du modèle de récompense, le score peut monter tandis que la qualité humaine baisse : il faut des évaluations tenues à l'écart et des contrôles de sur-optimisation.

Construction des préférences humaines

Pour une réponse préférée y_w et une réponse rejetée y_l :

$$\mathcal{L}_{\mathrm{RM}}(\phi) = -\mathbb{E} \log \sigma(r_\phi(x, y_w) - r_\phi(x, y_l))$$

Le modèle de récompense apprend un ordre relatif, pas une mesure absolue universelle.

Protocole d'annotation

- critères séparés : exactitude, utilité, style, sécurité ;
- randomisation de l'ordre ;
- options égalité / non-évaluable ;
- mesure d'accord ;
- audit par sous-groupes d'annotateurs et de prompts.

PPO : boucle d'optimisation

1. échantillonner des réponses avec la politique courante $\pi_{\theta_{\text{old}}}$;
2. calculer un signal de récompense et une pénalité de divergence à la référence ;
3. estimer les avantages A_t à partir des retours ;
4. effectuer plusieurs mises à jour sur le lot collecté, sous contrainte de proximité ;
5. rééchantillonner avec la nouvelle politique.

Objectif régularisé typique

$$\max_{\theta} \mathbb{E}_{y\sim\pi_\theta(\cdot|x)} \left[r_\phi(x,y) - \beta D_{\text{KL}}(\pi_\theta(\cdot \mid x) \,\|\, \pi_{\text{ref}}(\cdot \mid x)) \right].$$

PPO est on-policy : réutiliser trop longtemps les réponses d'une ancienne politique augmente le décalage de distribution.

PPO : objectif écrêté

Pour les tokens échantillonnés par $\pi_{\theta_{\text{old}}}$, définir

$$\rho_t(\theta) = \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)}.$$

L'objectif substitutif est

$$\mathcal{L}^{\text{clip}}(\theta) = \mathbb{E}_t[\min(\rho_t(\theta)A_t,\ \text{clip}(\rho_t(\theta), 1-\epsilon, 1+\epsilon)A_t)]\,.$$

- $A_t > 0$ favorise l'action ; $A_t < 0$ la défavorise ;
- l'écrêtage limite le gain procuré par une variation excessive du ratio ;
- il ne borne pas directement la divergence KL totale : celle-ci doit être suivie séparément.

Source : Schulman et al., 2017.

Direct Preference Optimization (DPO)

Idée

DPO part d'un objectif de récompense régularisé par une divergence KL et d'un modèle de préférences de Bradley–Terry. Il en déduit une perte qui optimise directement une politique sur des triplets (x, y_w, y_l), relativement à une politique de référence.

- pas de modèle de récompense séparé ;
- pas de collecte on-policy dans une boucle PPO ;
- objectif de classification binaire différentiable.

Source : Rafailov et al., 2023.

Limites de DPO

- préférences bruitées ou contradictoires ;
- prompts hors distribution ;
- biais de longueur et de style ;
- données de préférences peu représentatives ;
- choix de β et de la référence.

Objectif de DPO

L'Optimisation Directe des Préférences (DPO) vise à aligner les modèles de langage avec les préférences humaines en optimisant directement une politique π par rapport à ces préférences.

Objectif DPO

$$\pi^*(y|x) = \frac{1}{Z(x)} \pi_{\text{ref}}(y|x) \exp\left(\frac{r(x,y)}{\beta}\right) \tag{3}$$

où :

- $\pi_{\text{ref}}(y|x)$ est la politique de référence.
- $r(x,y)$ est la fonction de récompense reflétant les préférences humaines.
- $Z(x)$ est la fonction de partition, assurant la normalisation.
- β est un paramètre de température contrôlant la contrainte de divergence KL.

Perte DPO effectivement optimisée

Pour un triplet (x, y_w, y_l), où y_w est préféré à y_l, DPO minimise :

$$\mathcal{L}_{\mathrm{DPO}}(\theta) = -\mathbb{E}\left[\log \sigma\left(\beta \log \frac{\pi_\theta(y_w \mid x)}{\pi_{\mathrm{ref}}(y_w \mid x)} - \beta \log \frac{\pi_\theta(y_l \mid x)}{\pi_{\mathrm{ref}}(y_l \mid x)}\right)\right].$$

- π_{ref} ancre le modèle et pénalise les dérives.
- β règle la force relative de cette contrainte.
- Les préférences restent dépendantes du protocole d'annotation et de la population d'annotateurs.

DPO évite une boucle RL explicite, mais pas les problèmes de couverture, de bruit, de biais ni de sur-optimisation des préférences.

Source : Rafailov et al., 2023.

Bradley–Terry : modéliser une préférence binaire

Pour deux réponses y_w et y_l au même prompt x :

$$p(y_w \succ y_l \mid x) = \frac{\exp r(x, y_w)}{\exp r(x, y_w) + \exp r(x, y_l)} = \sigma(r(x, y_w) - r(x, y_l)) \,.$$

La log-vraisemblance d'une paire observée est donc

$$\log p(y_w \succ y_l \mid x) = \log \sigma(r_w - r_l) \,.$$

Invariance
Ajouter la même constante aux deux scores ne change pas la préférence.

Hypothèse
Une unique différence scalaire doit résumer le jugement comparatif conditionnellement à x.

Source : Bradley & Terry, 1952.

DPO : élimination de la fonction de partition

L'optimum de l'objectif récompense–KL vérifie

$$\pi^*(y \mid x) = \frac{1}{Z(x)} \pi_{\text{ref}}(y \mid x) \exp\left(\frac{r(x,y)}{\beta}\right).$$

Donc

$$r(x,y) = \beta \log \frac{\pi^*(y \mid x)}{\pi_{\text{ref}}(y \mid x)} + \beta \log Z(x).$$

Dans la différence $r(x, y_w) - r(x, y_l)$, le terme $\beta \log Z(x)$ s'annule. Bradley–Terry donne alors

$$p^*(y_w \succ y_l \mid x) = \sigma\left[\beta \log \frac{\pi^*(y_w \mid x)}{\pi_{\text{ref}}(y_w \mid x)} - \beta \log \frac{\pi^*(y_l \mid x)}{\pi_{\text{ref}}(y_l \mid x)}\right].$$

Cette annulation permet d'estimer la politique sans calculer $Z(x)$, mais elle reste conditionnée par le modèle de préférence choisi.

Questions de synthèse

1. Comment la tokenisation modifie-t-elle le coût et les erreurs d'un modèle ?
2. Quand une représentation TF–IDF reste-t-elle préférable à un modèle pré-entraîné ?
3. Pourquoi macro-F_1 et micro-F_1 peuvent-ils conduire à des conclusions opposées ?
4. D'où viennent la disparition et l'explosion du gradient dans un RNN ?
5. Quel rôle exact joue le masque causal dans un décodeur Transformer ?
6. Pourquoi l'apprentissage en contexte n'est-il pas un fine-tuning ?

Exigence

Répondre avec hypothèses, équations ou contre-exemple — pas par une simple définition.

RÉFÉRENCES

Articles et ressources de référence

Références : fondements et architectures

- Mikolov et al. (2013). *Distributed Representations of Words and Phrases and their Compositionality*. NeurIPS.
- Sennrich, Haddow & Birch (2016). *Neural Machine Translation of Rare Words with Subword Units*. ACL.
- Hochreiter & Schmidhuber (1997). *Long Short-Term Memory*. Neural Computation.
- Bahdanau, Cho & Bengio (2015). *Neural Machine Translation by Jointly Learning to Align and Translate*. ICLR.
- Vaswani et al. (2017). *Attention Is All You Need*. NeurIPS.
- Devlin et al. (2019). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. NAACL.
- Dao et al. (2022). *FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness*. NeurIPS.

Références : adaptation et alignement

- Brown et al. (2020). *Language Models are Few-Shot Learners*. NeurIPS.
- Ouyang et al. (2022). *Training Language Models to Follow Instructions with Human Feedback*. NeurIPS.
- Hu et al. (2022). *LoRA: Low-Rank Adaptation of Large Language Models*. ICLR.
- Dettmers et al. (2023). *QLoRA: Efficient Finetuning of Quantized LLMs*. NeurIPS.
- Rafailov et al. (2023). *Direct Preference Optimization: Your Language Model is Secretly a Reward Model*. NeurIPS.
- Jain & Wallace (2019). *Attention is not Explanation*. NAACL.
- Schulman et al. (2017). *Proximal Policy Optimization Algorithms*. arXiv.