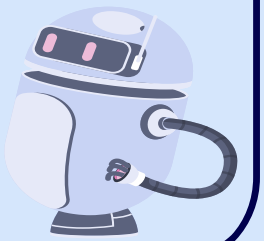
INTRODUCTION AUX OUTILS DU NLP

Ivanhoé Botcazou : i.botcazou@gmx.fr

Lundi 9 Février 2026



SOMMAIRE :

01

LES DONNÉES TEXTUELLES

Comment traiter des données non structurées ?

02

NOTION D'EMBEDDING

Comment donner du sens mathématique aux mots ?

03

LES MODÈLES TRANSFORMERS

Quels modèles pour générer des données textuelles ?

01 DES DONNÉES TEXTUELLES

Normaliser puis tokenizer les données

EXEMPLES DE TÂCHES POUR LE NLP

	title	text	label
0	Palestinians switch off Christmas lights in Be...	RAMALLAH, West Bank (Reuters) - Palestinians s...	1
1	China says Trump call with Taiwan president wo...	BEIJING (Reuters) - U.S. President-elect Donal...	1
2	FAIL! The Trump Organization's Credit Score W...	While the controversy over Trump s personal ta...	0
3	Zimbabwe military chief's China trip was norma...	BEIJING (Reuters) - A trip to Beijing last wee...	1
4	THE MOST UNCOURAGEOUS PRESIDENT EVER Receives ...	There has never been a more UNCOURAGEOUS perso...	0
...	...	...	...
24348	Mexico Senate committee OK's air transport dea...	MEXICO CITY (Reuters) - A key committee in Mex...	1
24349	BREAKING: HILLARY CLINTON'S STATE DEPARTMENT G...	IF SHE S NOT TOAST NOW THEN WE RE IN BIGGER TR...	0
24350	trump breaks from stump speech to admire beaut...	kremlin nato was created for agression \nruss...	0
24351	NFL PLAYER Delivers Courageous Message: Stop B...	Dallas Cowboys star wide receiver Dez Bryant t...	0
24352	NORDSTROM STOCK TAKES NOSEDIVE After Trump Twe...	UPDATE: Nordstrom stock closed up slightly tod...	0

Classification des fausses nouvelles sur plus de 45 000 articles de presse. Ces articles sont classés comme vrais (1) ou faux (0) :

(CNN) -- The company was founded in 1985 by seven communications industry veterans -- Franklin Antonio, Adelia Coffman, Andrew Cohen, Klein Gilhousen, Irwin Jacobs, Andrew Viterbi and Harvey White. One of Qualcomm's first products was OmniTRACS, introduced in 1988, which is currently the largest satellite-based commercial mobile system for the transportation industry. Today, Qualcomm's patent portfolio includes approximately 6,100 United States patents and patent applications for CDMA and related technologies. More than 130 telecommunications equipment manufacturers worldwide have licensed QUALCOMM's essential CDMA patents. Qualcomm is among the members of the S&P 500 Index, Fortune 500, and a winner of the U.S. Department of Labor's" Secretary of Labor's Opportunity Award." The company has been listed among Fortune's "100 Best Companies to Work For in America" for nine years in a row and the magazine's list of" Most Admired Companies." Qualcomm's Annual revenue for 2006 was $7.53 billion, with a net income of $2.47 billion. E-mail to a friend .	The company has become a huge name in communications in just 20 years . Qualcomm has a portfolio of approximately 6,100 U.S. patents . Fortune lists the company as one of the 100 best places to work in the U.S.

CNN/Daily Mail est un ensemble de données pour la synthèse de textes. Au total, le corpus comprend 286 817 paires d'entraînement,

NLP : Natural Language Processing (Traitement automatique du langage naturel)

translation
translation

{ "de": "Wiederaufnahme der Sitzungsperiode", "en": "Resumption of the session" }

{ "de": "Ich erkläre die am Freitag, dem 15. Dezember 2000, unterbrochene Sitzungsperiode des Europäischen Parlaments für wieder aufgenommen.", "en": "I declare resumed the session of the European Parliament adjourned on Friday, 15 December 2000." }

{ "de": "Erklärungen der Präsidentin", "en": "Statements by the President" }

{ "de": "Werte Kolleginnen und Kollegen, wie Sie wissen, hat ein weiteres Erdbeben in Mittelamerika in dieser bereits mehrfach seit Beginn des zwanzigsten Jahrhunderts schwer getroffenen Region verheerenden Schaden angerichtet.", "en": "Ladies and gentlemen, on Saturday, as you know, an earthquake struck Central America once again,…

{ "de": "Die vorläufige, schreckliche Bilanz in El Salvador lautet zurzeit bereits: 350 Tote, 1 200 Vermisste, eine vollständig verwüstete Region und Tausende zerstörter Häuser im gesamten Land.", "en": "The latest, provisional, figures for victims in El Salvador are already very high. There are 350 people dead, 1 200 people missing, th…

{ "de": "Die Europäische Union hat schon jetzt ihre Solidarität unter Beweis gestellt, indem sie eine Hilfsmannschaft vor Ort geschickt hat, während die Bereitstellung von Finanzhilfen der EU und ihrer Mitgliedstaaten bereits erfolgte oder in Kürze erfolgen wird. Ich kann Ihnen mitteilen, dass einige Fraktionen unseres Parlaments…

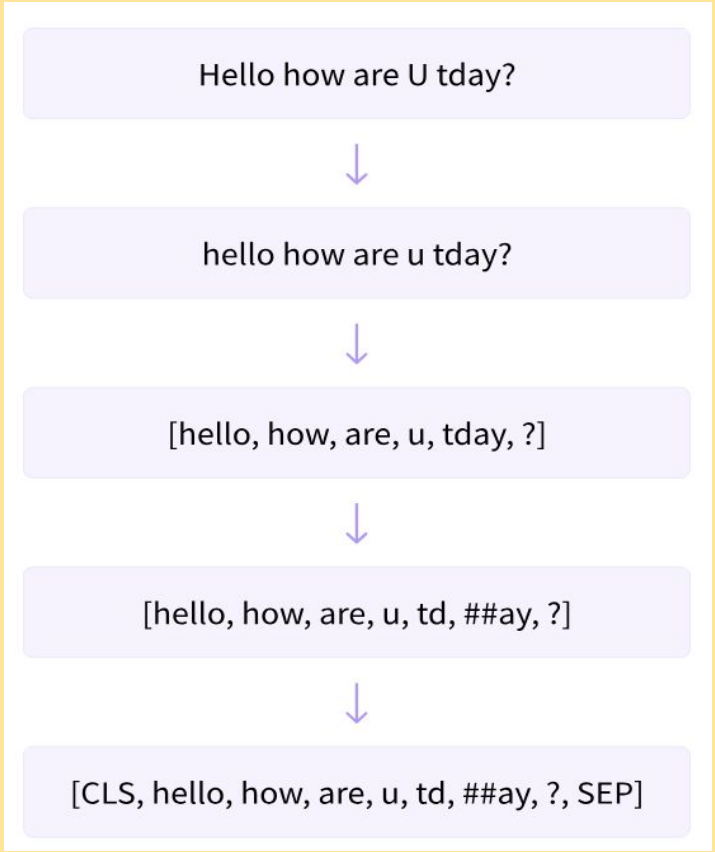
{ "de": "Des Weiteren möchte ich Sie darüber in Kenntnis setzen, dass ich dem Präsidenten von El Salvador - natürlich im Namen des Europäischen Parlaments - unser Beileid übermittelt und angesichts der Tragödie, die dieses Land durchlebt, unser tiefstes Mitgefühl zum Ausdruck gebracht habe.", "en": "However, I should like to…

{ "de": "Aus Achtung vor den Opfern und dem unermesslichen Leid ihrer Familien möchte ich Sie bitten, eine Schweigeminute einzulegen.", "en": "I would ask you, as a mark of respect for the victims and for the immense suffering of their families, to observe a minute' s silence." }

WMT 2014 est un ensemble de données de référence couramment utilisé pour évaluer les modèles de langage développés pour la traduction automatique neuronale (NMT, pour *Neural Machine Translation*).

L'arrivée des modèles Transformers, introduits dans l'article **"Attention Is All You Need"** publié en 2017, a marqué un tournant en 2019 grâce aux performances obtenues sur ces ensembles de données de référence (*benchmark datasets*).

NORMALISATION ET TOKENIZATION



Texte brut

Normalisation

Pre-tokenization

Tokenization

Tokens spéciaux

Données brutes sans normalisation : 𝐻ello 𝐖örld, 𝐿et's ``nôrmälize´´ this sentençe

La normalisation des données joue un rôle essentiel dans la **réduction de la taille du vocabulaire**. Cette étape consiste à **uniformiser le texte** en appliquant diverses transformations :

- **Conversion en minuscules** : "CHAT" et "chat" deviennent identiques
- **Suppression des espaces superflus** : considéré comme du bruit dans les données textuelles.
- **Suppression des “stop words”** : ces mots (“a,” “the,” “is,” “are,” …) contiennent peu de sens sémantique et peuvent parfois être retirés selon les tâches de NLP.

```
tokenizer = AutoTokenizerFast.from_pretrained('…')

Text_normalized = tokenizer.backend_tokenizer.normalizer.normalize_str(text)
```

Les *tokenizers* issus de la classe **AutoTokenizerFast** du module Hugging Face disposent de leur propre algorithme de normalisation. Un *tokenizer* est spécifiquement associé à un modèle de langage, ce qui garantit une compatibilité optimale entre les deux.

𝒽ℯll⓪ 🅦örld, 𝓛et's ``nôrmälize´´ this sentençe

```
XLMRobertaTokenizerFast.from_pretrained("xlm-roberta-base")
```

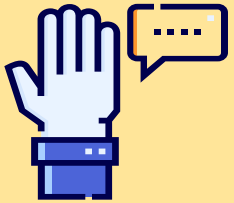
Hello 🅦örld, Let's ``nôrmälize´´ this sentençe

```
FNetTokenizerFast.from_pretrained("google/fnet-base")
```

𝒽ℯll⓪ 🅦örld, 𝓛et's ``nôrmälize´´ this sentençe

```
RetriBertTokenizerFast.from_pretrained("yjernite/retribert-base-uncased")
```

𝒽ℯll⓪ 🅦orld, 𝓛et's ``normalize´´ this sentence



La suppression des accents peut conduire à des **ambiguïtés** dans le texte, surtout dans les langues où les accents modifient la grammaire ou le sens des mots.

Vérifier que la **normalisation** n'a pas introduit des erreurs dans les données, notamment en effectuant un contrôle qualité sur un échantillon des données prétraitées.

An indignant father

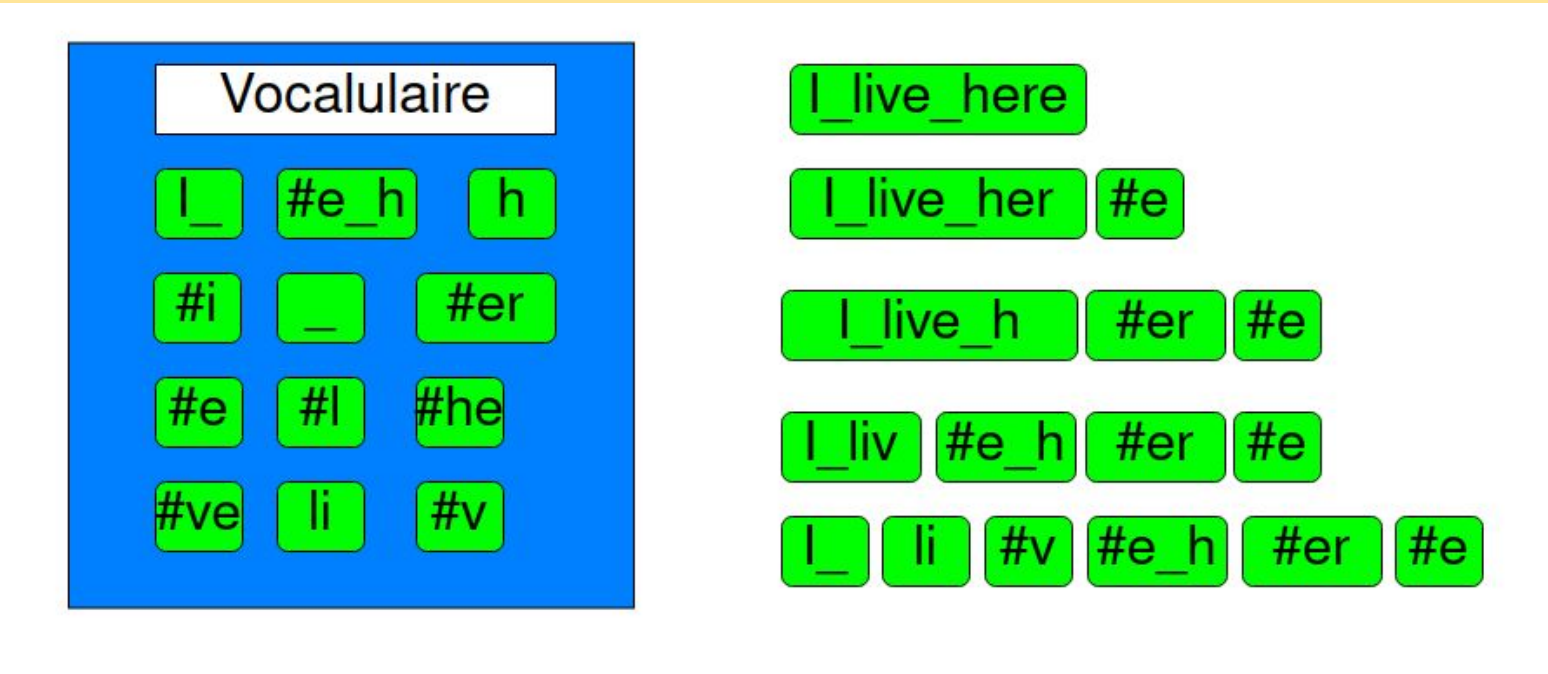
```
text = “un père indigné"

tokenizer = AutoTokenizerFast.from_pretrained('distilbert-base-uncased')


print(tokenizer.backend_tokenizer.normalizer.normalize_str(text))
```

```
'un pere indigne'
```

An unworthy father

LA TOKENIZATION

La **tokenization** consiste à découper les données textuelles en sous-unités de longueurs variables selon un vocabulaire de référence appris durant une phase d'apprentissage. Chaque sous-unité textuelle est appelé **un token**. Un token peut être un mot, un groupe de mot ou encore quelques lettres/caractères.



L'**entrainement d'un tokenizer** sur un corpus de référence consiste à créer un **vocabulaire** en associant à chaque unité textuelle (mot, lettre, séquence de caractères, ...) un identifiant numérique.

Vocabulaire = {"a" : 1, "the" : 2, "u.s.a" : 3,"table": 4, ···, "?" :34_000 }

La taille du vocabulaire est fixe, en général elle est de l'ordre de **30K à 250K tokens**

```
tokenizer.get_vocab()
```

```
'burgl': 18115,
'douchebag': 22833,
'consort': 12876,
'rahm': 16771,
'compassionate': 17015,
'eat': 6397,
'guess': 4770,
'obligations': 7974,
'2009': 3611,
'merr': 13869,
'severed': 15602,
'#': 2,
```

content	ids	tokens
WATCH: This Priceless Video HILARIOUSLY Exposes Trump And His Doctor For What They Really Are Trust Funny Or Die to come up with something like this. They ve released a video starring Brent Spiner, a.k.a Dr. Okun from Independence Day, and Data from Star Trek: The Next Generation, and he (ahem) takes on a new role here to lampoon the hell out of both Trump and his quack of a doctor.Spiner actually introduces himself as Dr. Bornstein, and immediately launches into what he knows about Trump: I m making this video because there s something they don t want you to know. Trump s not from here! And I don t mean not from America, I mean not from this planet, okay? When I wrote that ridiculously glowing letter in five minutes, I thought you d see through it for the obvious farce that it was! Indeed, that letter has been the subject of controversy for Trump, seeing as how Bornstein used language that doctors generally don t use in it. He said that Trump s lab results were astonishingly excellent, for one thing. He also declared Trump the healthiest person ever to run for president, which is something he can t possibly know.Spiner hilariously touches on all of that, too, in his role as the infamous Dr. Bornstein. He then goes back into discussing Trump s alien anatomy: I did a chest exam, and to be honest, I was not expecting to find a heart, and guess what? I didn t! What Bornstein did find in place of a heart is even funnier than Trump s obvious lack of one. From there, it just spirals. Watch below to find out what else Bornstein knows about Trump:	[857, 25, 230, 13255, 663, 9455, 10372, 168, 123, 202, 5801, 136, 383, 246, 1152, 185, 2634, 5432, 109, 2675, 113, 503, 286, 181, 1256, 407, 230, 13, 246, 137, 1574, 38, 663, 19718, 19373, 368, 4743, 11, 38, 13, 48, 13, 38, 507, 13, 2254, 152, 225, 2200, 228, 11, 123, 2059, 225, 1768, 16989, 25, 105, 949, 4975, 11, 123, 135, 7, 38, 7925, 8, 2699, 104, 38, 263, 1732, 723, 113, 4652, 21909, 105, 3155, 192, 122, 875, 168, 123, 202, 276, 232, 122, 38, 5801, 13, 368, 4743, 1398, 18911, 1373, 120, 507, 13, 2727, 2861, ...]	[watch, :, this, priceless, video, hilariously, exposes, trump, and, his, doctor, for, what, they, really, are, trust, funny, or, die, to, come, up, with, something, like, this, ., they, ve, released, a, video, starring, brent, sp, iner, ,, a, ., k, ., a, dr, ., ok, un, from, independence, day, ,, and, data, from, star, trek, :, the, next, generation, ,, and, he, (, a, hem,), takes, on, a, new, role, here, to, lam, poon, the, hell, out, of, both, trump, and, his, qu, ack, of, a, doctor, ., sp, iner, actually, introduces, himself, as, dr, ., born, stein, ...]

Idée Naïve : la tokenisation basée sur les espaces et la ponctuation ne gère pas correctement les expressions spécifiques ou les constructions linguistiques complexes :

- Les contractions : *"I'm"* pourrait être divisé en *"I"* et *"m"*.
- Les acronymes : *"E.U."* ou *"N.A.S.A"* pourrait être morcelés comme *"E"*, *"U"*, et *"."*
- Les entités composées : *"New York"* pourrait être séparé en *"New"* et *"York"*, perdant ainsi son sens d'entité unique.

Tokenisation basée sur les sous-mots :

Les algorithmes de tokenization comme ***Byte Pair Encoding*** (BPE) ou ***WordPiece*** segmentent les mots rares en unités plus fréquentes, permettent de capturer des constructions comme *"unthinkable"* (découpé en *"un"*, *"#think"*, *"#able"*) tout en réduisant la taille du vocabulaire. Ces algorithmes se basent sur une approche fréquentielle d’apparition des mots.

Algorithm : Byte Pair Encoding (BPE)

1. Initialisation

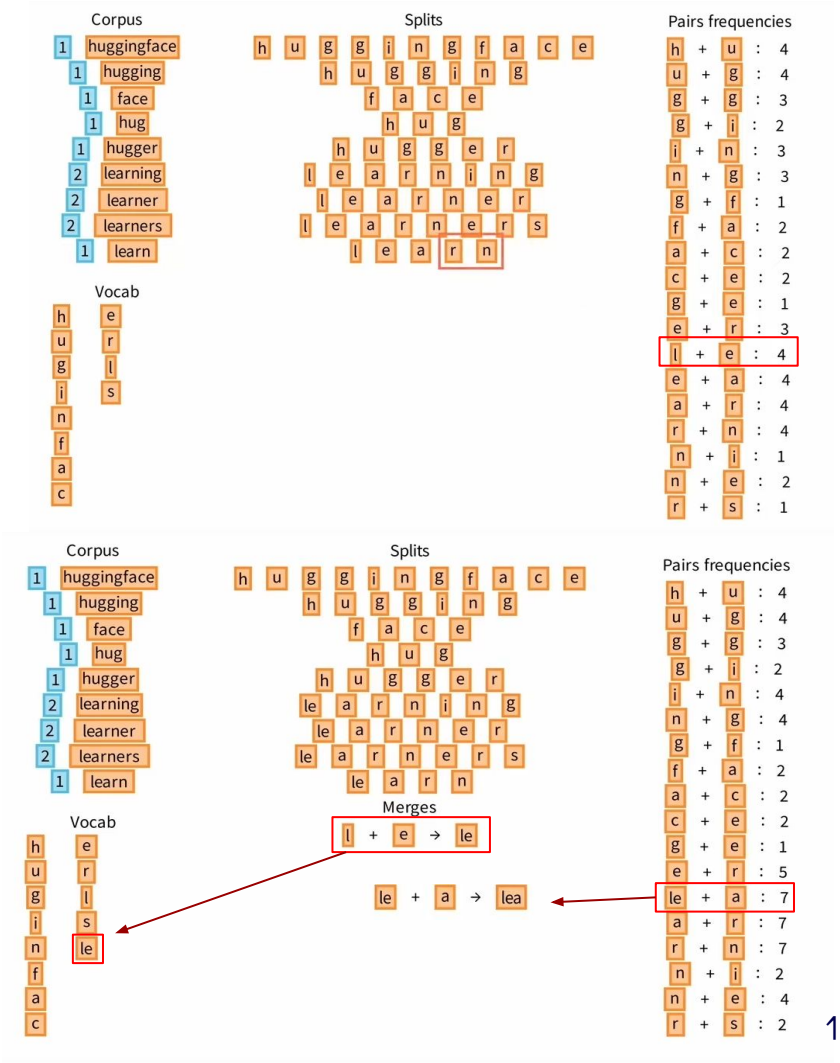
- Tous les caractères Unicode sont identifiés et listés comme **tokens initiaux** du vocabulaire.

2. Itération

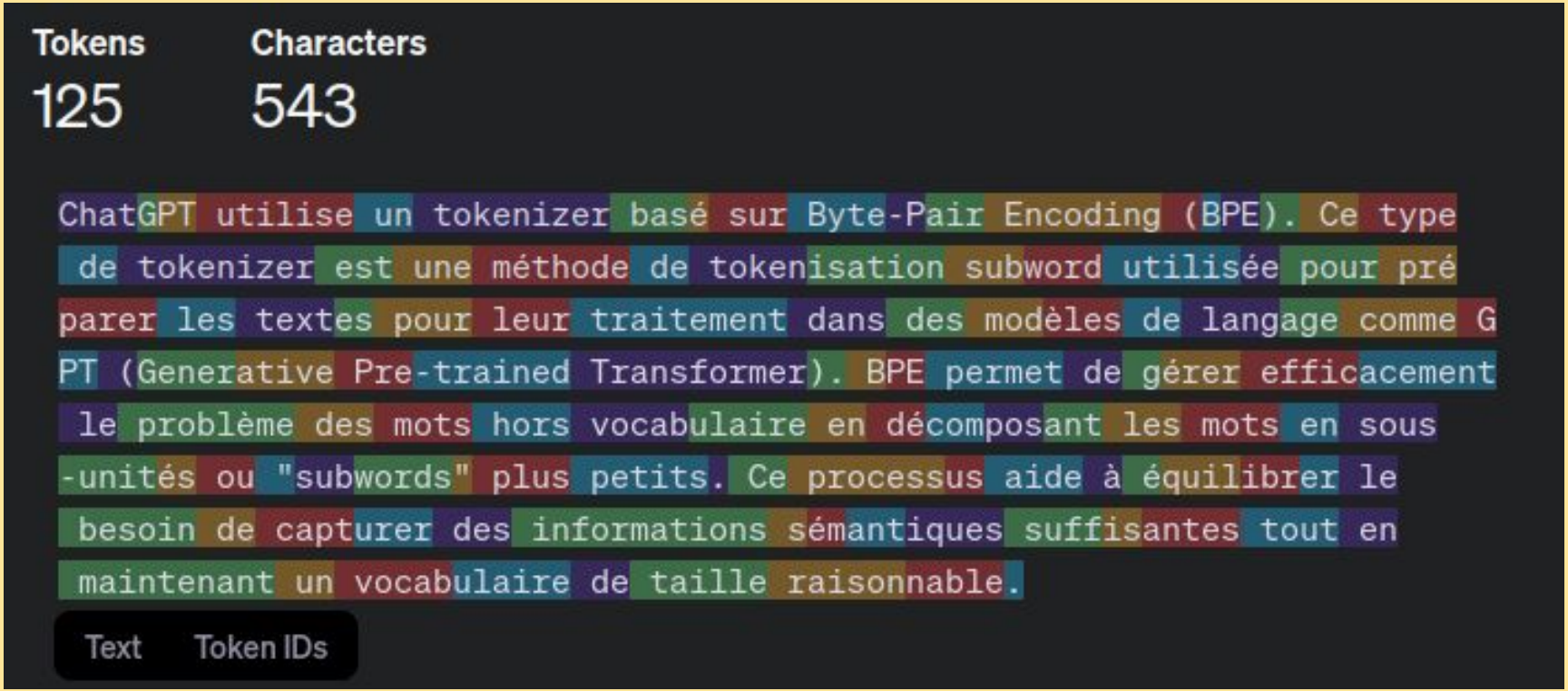
- **Étape 1** : identifier la **paire de tokens la plus fréquente** (bigramme) dans le corpus.
- **Étape 2** : fusion des deux tokens pour créer un **nouveau token**.
- **Étape 3** : ajouter ce token au **dictionnaire**.

3. Finalisation

- Le processus continue jusqu'à ce que la **taille du vocabulaire** atteigne sa **limite prédéfinie**.
- Résultat : un vocabulaire incluant à la fois des **mots entiers** et des **sous-mots**.



Tokenisation de la réponse de GPT-4 en lien avec le tokenizer qu'il utilise.



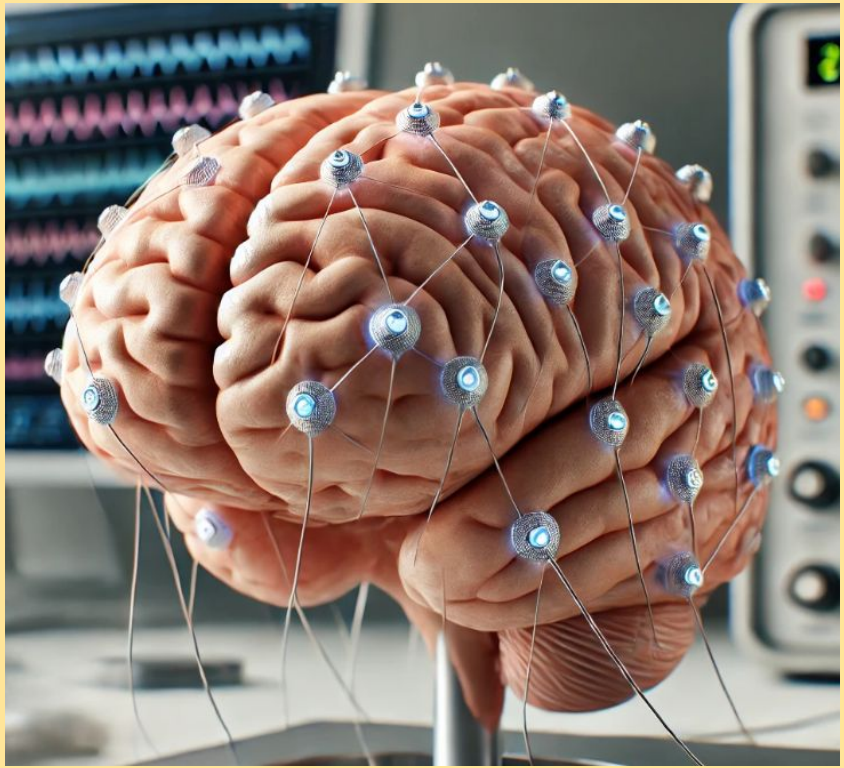
https://platform.openai.com/tokenizer

02

NOTION D’ EMBEDDING

Représenter mathématiquement les tokens

VECTORISATION DES TOKENS : EMBEDDING



Une expérience fictive pour comprendre les embeddings :

- Placer des électrodes sur le crâne de plusieurs patients.
- Mesure de l'activité des zones cérébrales après une stimulation auditive.
- Analyse des réponses à l'écoute de certains mots.
- Réponses électriques seraient traduites en vecteurs dans un espace vectoriel multidimensionnel.

accident	car	caught	fire	jam	kind	sadly	set	swear	true
0.00	0.00	0.0	0.0	0.00	0.67	0.53	0.00	0.00	0.53
0.00	0.00	0.0	0.0	0.47	0.00	0.00	0.47	0.37	0.00
0.59	0.47	0.0	0.0	0.00	0.00	0.00	0.00	0.47	0.47
0.00	0.64	0.4	0.4	0.00	0.00	0.32	0.00	0.00	0.00

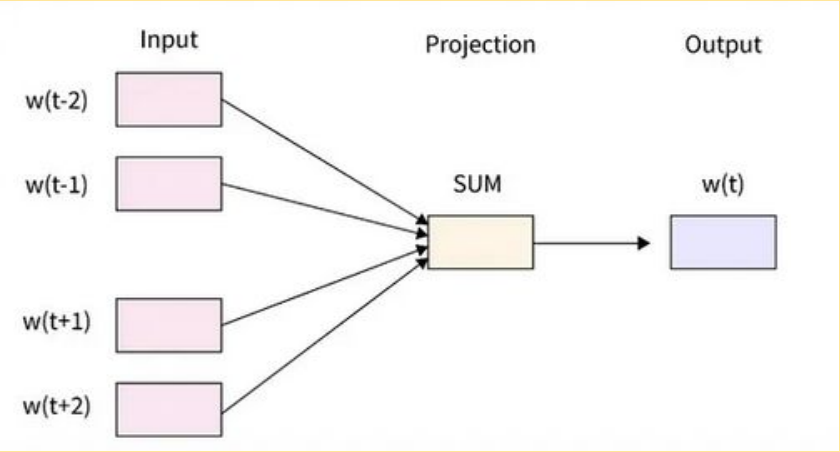
Table d’embeddings : chaque token du vocabulaire est associé à un vecteur qui contiendrait son sens sémantique. La dimension des vecteurs sera notée ***d_model***.

EXEMPLE D'EMBEDDINGS : WORD2VEC

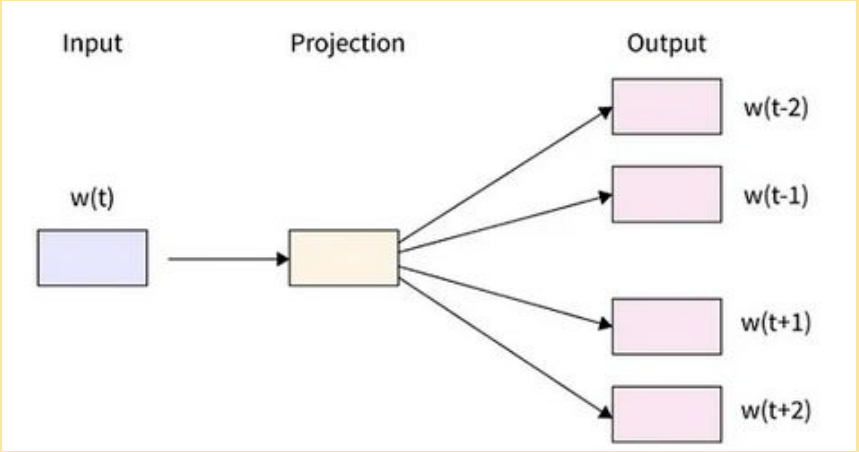
***Idée* : Construire une table embeddings en extrayant les poids d'un réseau de neurones entraîné sur une tâche de classification.**

Deux modèles de classification proposés :

- **Continuous Bag of Words (CBOW)** : prédire un mot cible à l'aide du contexte des mots environnants.
- **Skip-Gram :** prédire les mots environnants de contexte à l'aide d'un mot en entrée.



Continuous Bag of Words (CBOW)

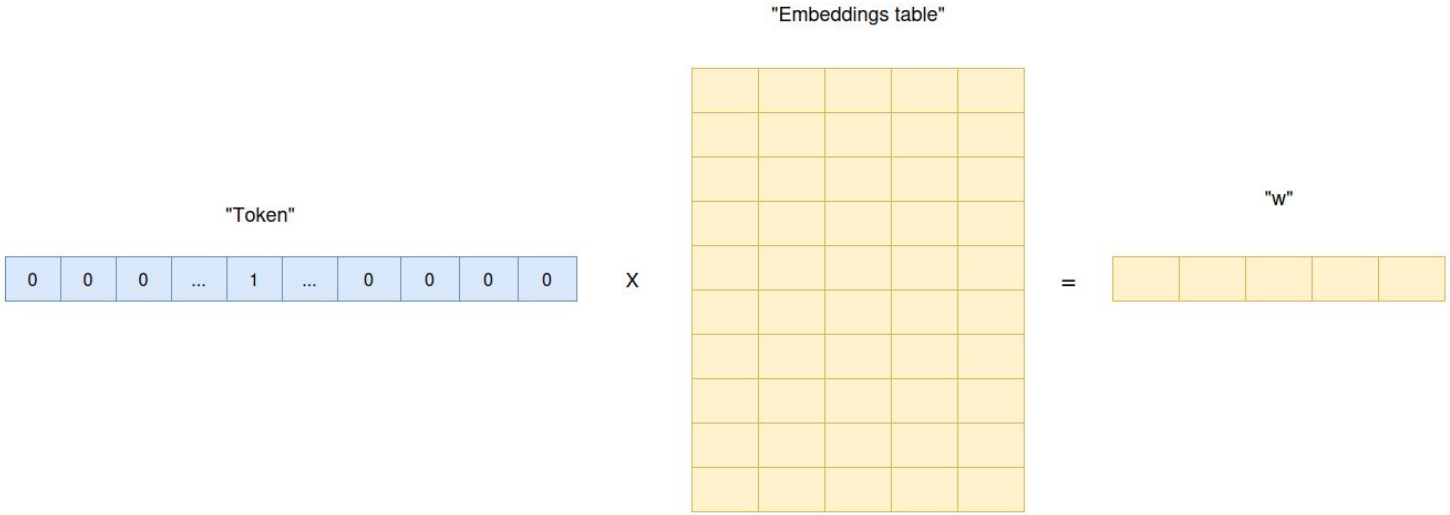


Skip-Gram

Entraîner les poids de la table d'embeddings pour Word2Vec

Soit [Token1,Token2,…,TokenN] une famille de tokens issu du corpus d'entraînement (ex: Wikipedia) :

- **Représentation en "One-Hot Encoding" :** équivalent à un Dirac en la position du token T_i dans le vocabulaire du Tokenizer.
- **Représentation vectorielle “w”** de taille d_model, obtenue par le produit avec la table d'embeddings.



La table d’embeddings est initialisée de manière aléatoire, elle se forge au cours de l’apprentissage.

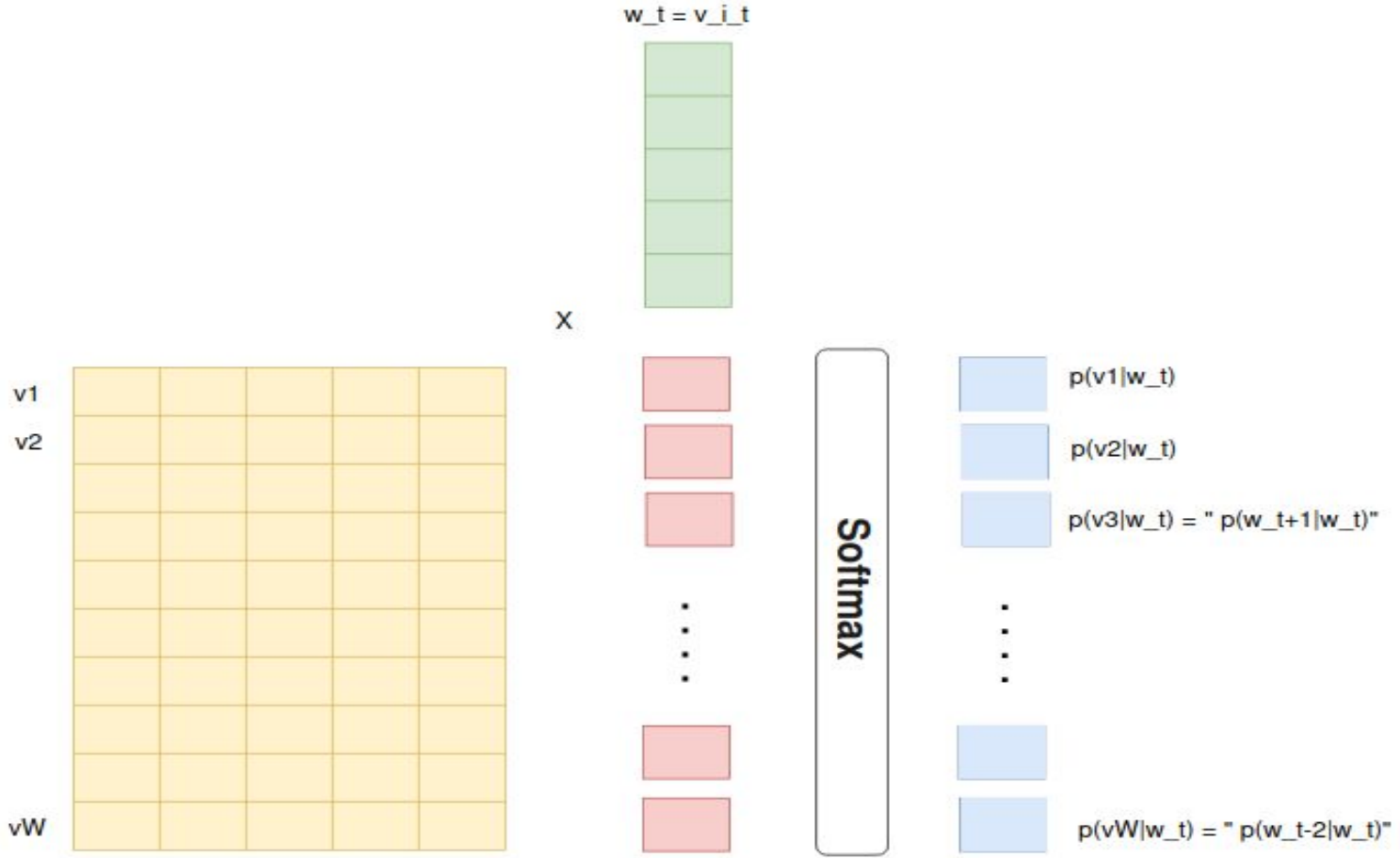
Idéalement, pour une séquence de tokens vectorisés $w_1^\theta, w_2^\theta, \ldots, w_N^\theta$ issue des données d'entraînement, l'objectif du modèle Skip-Gram serait de minimiser une $\mathcal{L}$oss en les poids θ :

$$\mathcal{L}(\theta) = \frac{1}{N-2c} \sum_{t=1+c}^{N-c} \sum_{\substack{-c \leq j \leq c \\ j \neq 0}} -\log p(w_{t+j}^\theta \mid w_t^\theta)$$

où c est la taille du contexte, w_t^θ est le vecteur associé au token central et une fonction Softmax donne la probabilité suivante :

$$p(w \mid w') = \frac{\exp\left(w^\top w'\right)}{\sum_{i=1}^{W} \exp\left(w^\top v_i\right)}$$

où W la taille du vocabulaire et v_i est la représentation vectorielle du token i dans le vocabulaire.



Les poids de la table d'embeddings sont entraînés via une tâche de classification sur tous les mots du vocabulaire.

La fonction de perte à minimiser est l'***entropie croisée***.

Soit $I = \{i_{t-c}, \cdots, i_{t-1}, i_t, i_{t+1}, \cdots, i_{t+c}\}$ l'ensemble des indices associés à la séquence de tokens vectorisés $\{w_{t-c}, \cdots, w_{t-1}, w_t, w_{t+1}, \cdots, w_{t+c}\}$

$$\sum_{\substack{-c \leq j \leq c \\ j \neq 0}} -\log p(w_{t+j} \mid w_t) = \sum_{\substack{j \in I \\ j \neq i_t}} -\log p(v_j \mid v_{i_t})$$

$$= 2c \times \left(\sum_{j \in I \setminus \{i_t\}} -\frac{1}{2c} \times \log p(v_j \mid v_{i_t}) + \sum_{j \notin I \cup \{i_t\}} -0 \times \log p(v_j \mid v_{i_t}) \right)$$

$$= 2c \times \sum -\frac{1}{2c} \delta_{I \setminus \{i_t\}}(j) \times \log p(v_j \mid v_{i_t})$$

$$= 2c \times H\left(\frac{1}{2c} \delta_{I \setminus \{i_t\}} \mid p(\,\cdot \mid v_{i_t}) \right)$$

Il y a donc un lien avec la divergence de **Kullback–Leibler**

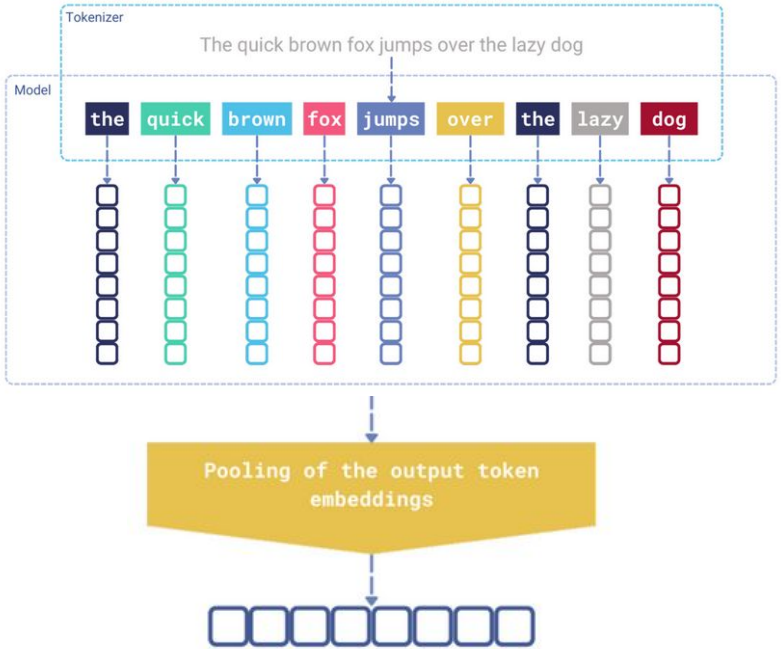
```
sentences = [
    ['this', 'is', 'an', 'example', 'sentence', 'for', 'word2vec'],
    ['we', 'are', 'creating', 'a', 'word2vec', 'model', 'using', 'the', 'gensim', 'library'],
    ['we', 'are', 'working', 'with', 'cbow', 'and', 'skipgram', 'models'],
    ['python', 'is', 'a', 'programming', 'language', 'for', 'natural', 'language', 'processing'],
    ['word2vec', 'is', 'one', 'of', 'the', 'word', 'embedding', 'techniques'],
    ['the', 'word2vec', 'model', 'is', 'used', 'for', 'word', 'embeddings'],
    ['gensim', 'provides', 'an', 'easy', 'way', 'to', 'train', 'word2vec', 'models'],
    ['many', 'researchers', 'use', 'word2vec', 'for', 'various', 'nlp', 'tasks'],
    ['the', 'skipgram', 'model', 'focuses', 'on', 'predicting', 'context', 'words'],
    ['cbow', 'model', 'predicts', 'the', 'center', 'word', 'from', 'context', 'words'],
    ['natural', 'language', 'processing', 'involves', 'working', 'with', 'large', 'datasets']
]
```

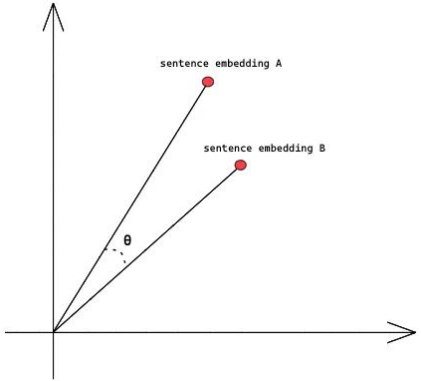
```
cbow_model = Word2Vec(sentences, vector_size=100, window=5, min_count=1, sg=0, epochs=100)
skipgram_model = Word2Vec(sentences, vector_size=100, window=5, min_count=1, sg=1, epochs=100)

cbow_model.train(sentences, total_examples=len(sentences), epochs=100)
skipgram_model.train(sentences, total_examples=len(sentences), epochs=100)
```

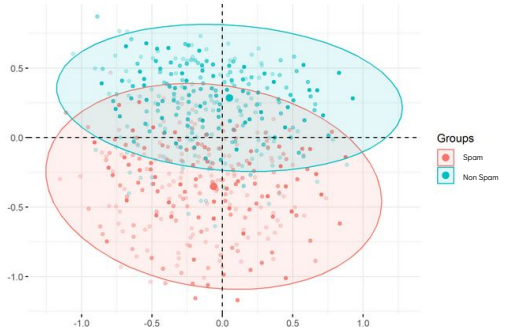
```
skipgram_model.wv.get_vector('word2vec')

# -> array([-0.00492096, -0.00097772,  0.05365122,  0.08740456, -0.09275784,
#       -0.07140654,  0.07582574,  0.09669966, -0.05774484, -0.03815871]
#       ,dtype=float32)
```

CLASSIFICATION AVEC EMBEDDINGS STATIQUES

Pour des tâches de classification, comme la détection de **spams** ou de ***fake news***, une approche naïve consiste à calculer le **vecteur barycentrique** des *embeddings* des *tokens*. Cette opération vectorielle permet de résumer l'information d'un texte en un vecteur global.







Ces représentations vectorielles des textes génèrent de nouvelles composantes, ou "**features**", qui peuvent être utilisées pour entraîner un modèle de machine learning dédié à des tâches de classification.

LES LIMITES DES EMBEDDINGS STATIQUES

- **Absence de prise en compte du contexte :**

  Les embeddings restent les mêmes, quelle que soit la variation contextuelle, ce qui peut entraîner des ambiguïtés, ces embeddings ne capture pas le changement de sens des mots selon le contexte.

  Exemple : *"Je mange avec un avocat"* (légal) **VS** *"Je mange un avocat"* (aliment).

- **Absence d'information spatiale et temporelle dans un texte :**

  Le pooling supprime la notion d'ordre des mots ainsi que leurs relations syntaxiques, ce qui peut conduire à une perte d'information critique.

  Exemple :*"le chat court après la souris" = "la souris court après le chat"*

- **Représentation statique et non adaptée aux séquences longues :**

  Le pooling dilue l'information, plus il y a d'éléments à agréger, plus la représentation globale perd en précision.

TP1 : CLASSIFICATION BINAIRE

Objectif : créer un classificateur performant pour la détection de faux articles d'actualité issus d'un ensemble de données *Kaggle*.

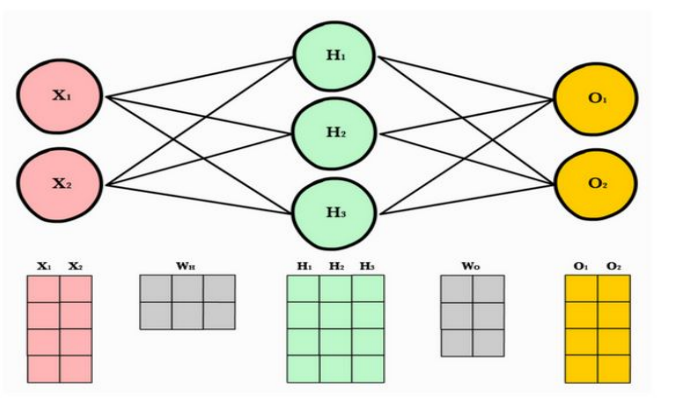
03

MÉCANISME D'ATTENTION

Les modèles Transformers pour avoir des embeddings dynamiques

RÉSEAUX DE NEURONES TRANSFORMERS

- Tout repose sur un agencement particulier de **réseaux de neurones** du type **multilayer perceptron (MLP) - Fully connected (FC).**

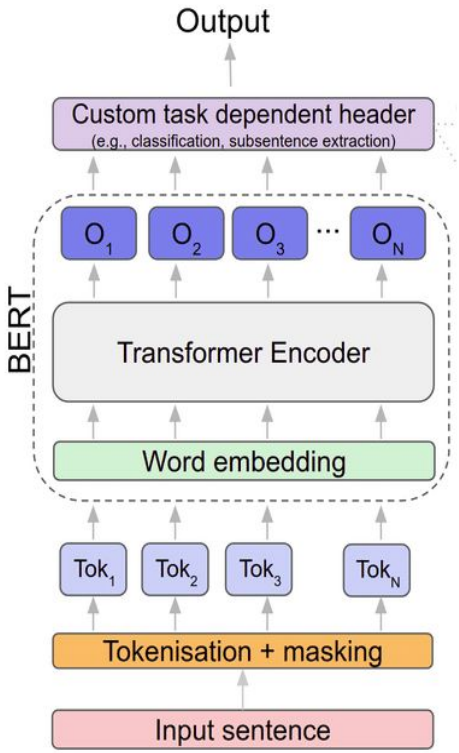


- Les **Transformers** partagent tous le principe fondamental de l'**attention**.
- L'apprentissage des poids du réseau se fait via des **tâches de classification**.

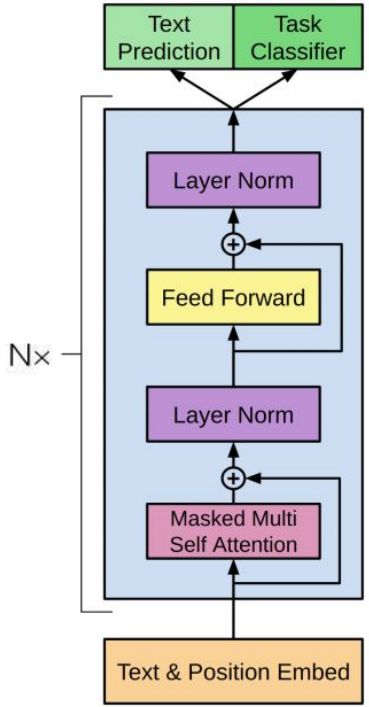
Les **réseaux de neurones transformers** se déclinent en trois catégories :

- **Encoder-decoder** encore appelé modèle **Seq2seq** initialement introduit dans l'article "**Attention is all you need**" la *Transformer-based* par *Vaswani et al*. Ex : **T5** (Google), **Bart** (Meta), **Pegasus** (Google), **ProphetNet** (microsoft) **…**
- **Encoder** caractérisé par une **"attention bidirectionnelle"** et sont souvent appelés modèles d'**auto-encodage**. Ex : **Bert**, **Roberta**, **DistilBert** …
- **Decoder** caractérisé par une **"attention causale".** Ces modèles sont souvent appelés modèles **auto-régressifs**. Ex : **GPT** (OpenAI), **LLama** (Meta), **GEMMA** (Google)

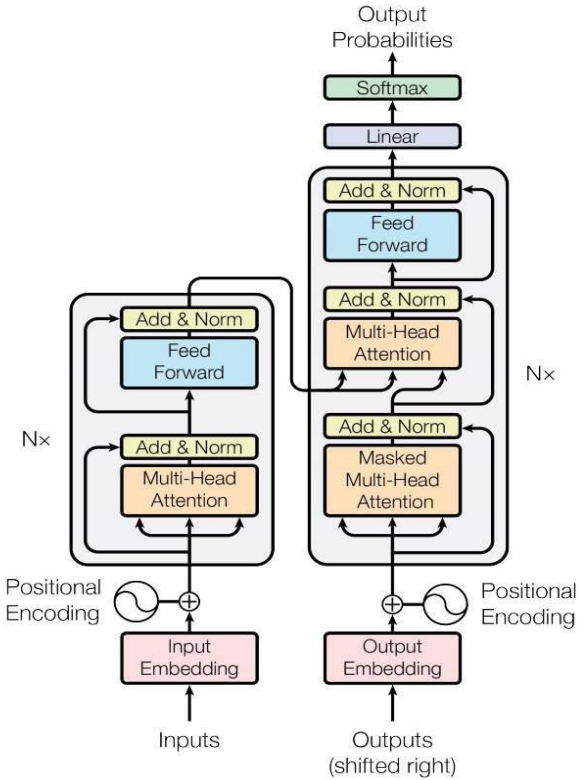
SCHÉMAS DES DIFFÉRENTES ARCHITECTURES



Modèle Encoder :



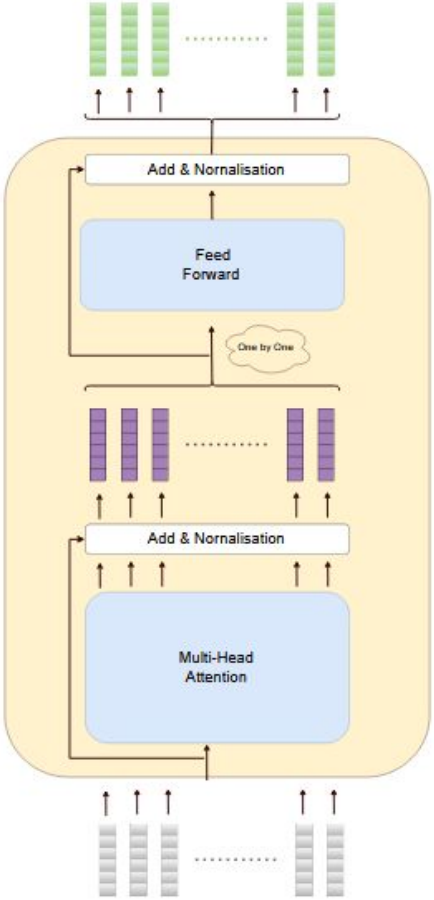
Modèle Decoder :



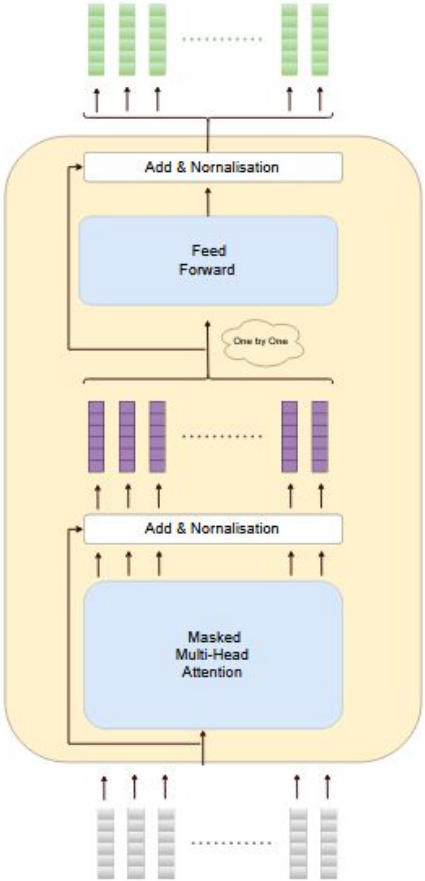
Modèle SeqToSeq :

Les blocs de Transformers manipulent des **séquences de vecteurs** et renvoient des séquences de vecteurs de même longueur.

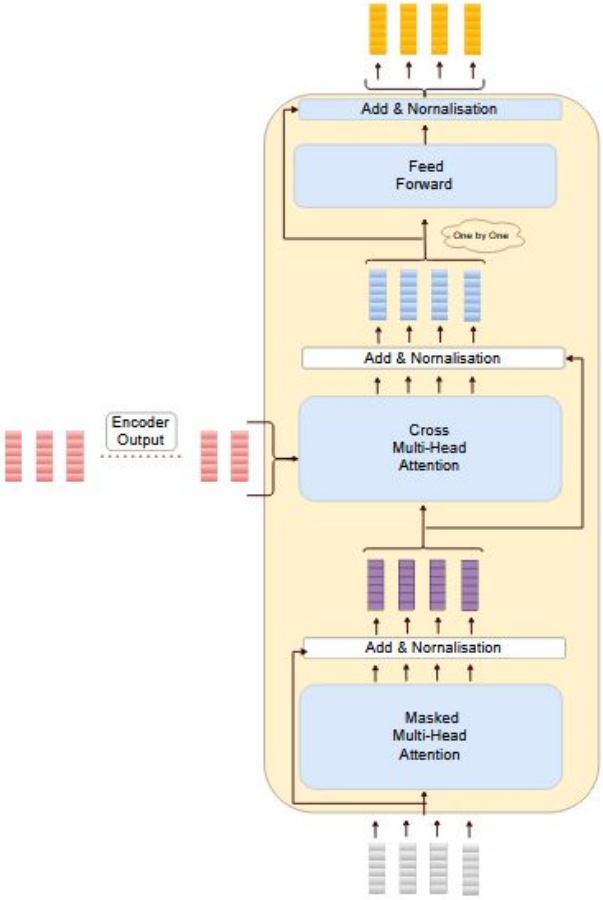
DIFFÉRENTS BLOCS DE TRANSFORMERS



Modèle Encoder :



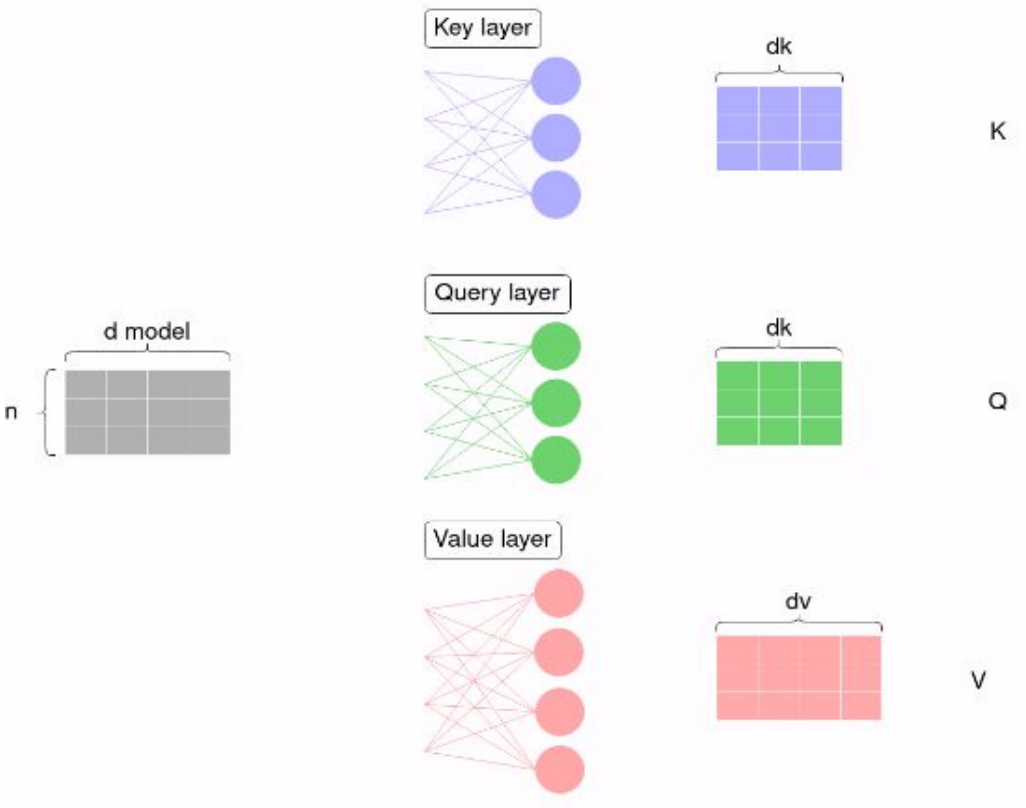
Modèle Decoder :



Modèle SeqToSeq :

MÉCANISME D'ATTENTION

- Traitement de la séquence des embeddings via trois réseaux de neurones linéaires : **Keys**, **Queries** et **Values**.
  - **d_model** est la dimension des embeddings. (Ex : 768 Bert, 4096 LLAMA.3.1.)
  - **n** est le nombre de tokens dans l'input, fenêtre de contexte. (Ex : 512 T Bert, 128K T LLAMA.3.1.)
  - **dk** et **dv** nombre de neurones dans les réseaux K, Q et V. (Ex : dk = dv = d_model)



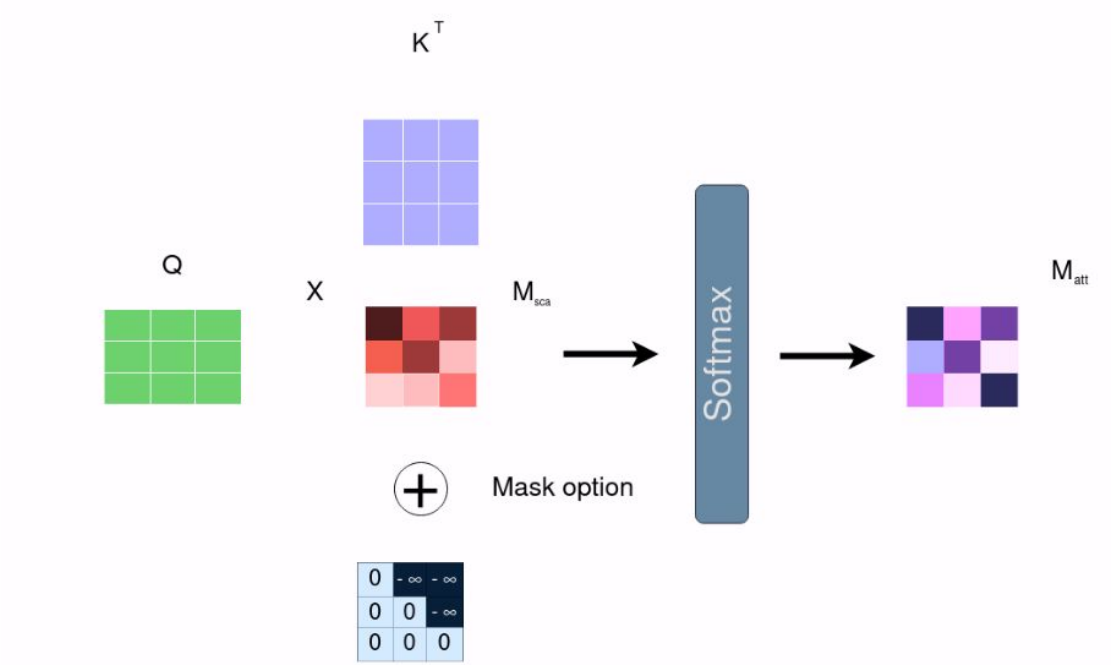
❖ Calcul de la **matrice d'attention**, **normalisation** et passage au **softmax sur les lignes** pour obtenir des pondérations probabilistes.

➢ Possibilité de rajouter un mask causal pour tuer les poids d'attention des tokens futurs en chaque position.

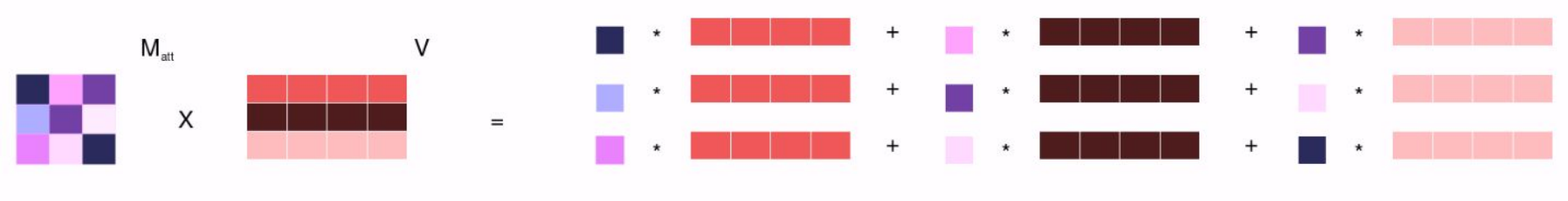
$$M_{\text{sca}} = \frac{QK^T}{\sqrt{d_k}}$$

$$M_{\text{att}} = Softmax(M_{\text{sca}}, axis = 1)$$

$$M_{\text{att}}(i,j) = \frac{\exp(M_{\text{sca}}(i,j))}{\sum_{k=1}^{n} \exp(M_{\text{sca}}(i,k))}$$



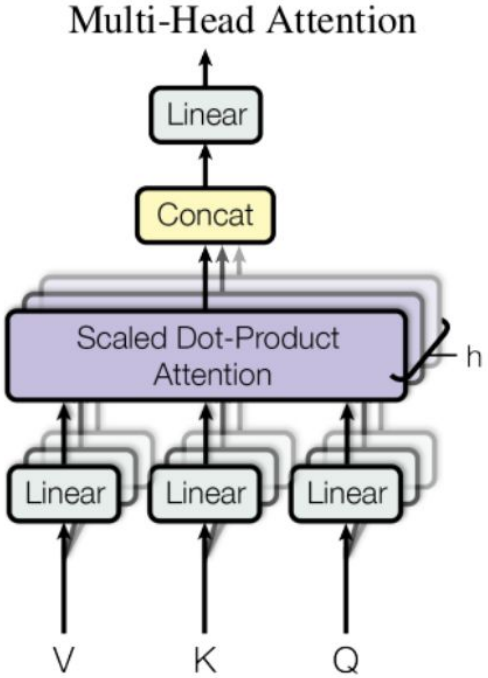
- Le produit matriciel entre la **matrice d'attention** le la matrice des **values** V donne pour chaque ligne de sortie une combinaison linéaire des lignes de la matrice V avec les poids de la matrice d'attention.



- Le mécanisme d'attention permet de représenter vectoriellement chaque token comme une pondération des autres tokens dans la séquence.

$$Attention(K, Q, V) = Softmax\left(\frac{QK^T}{\sqrt{d_k}}, axis = 1\right)V$$

- Le mécanisme d'attention est reproduit **en parallèle** dans plusieurs **têtes d'attention**.
  - Les sorties des têtes d'attention sont **concaténées** et ramenées à la dimension **d_model** par une couche dense.



$$MultiHead(Q,K,V) = Concat(head_1(K,Q,V),\cdots,head_h(K,Q,V)) \cdot W$$

- Dans papier original il y a 8 têtes d'attention par bloc.
  - Chaque tête possède une représentation du contexte et apporte une "expertise".
- Pensée pour paralléliser les calculs et ainsi optimiser les temps d'apprentissage.

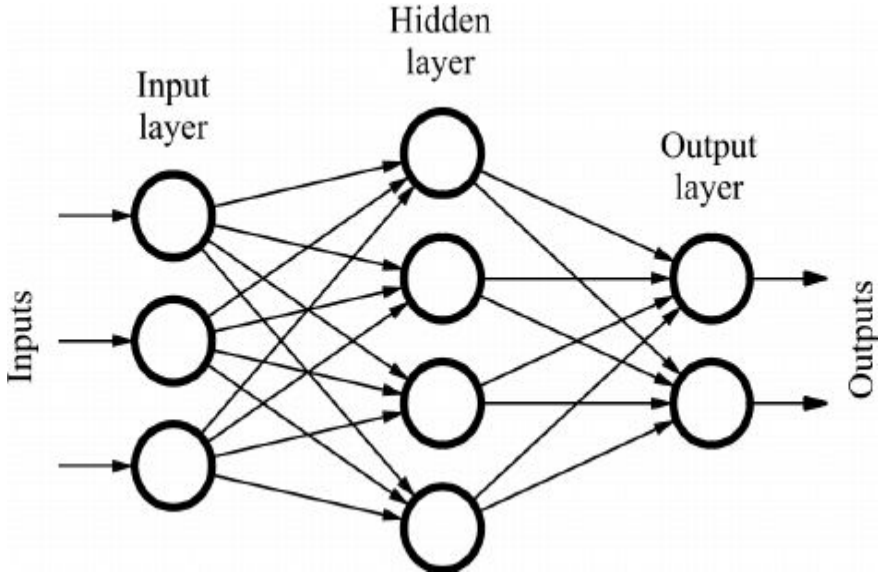
FEED FORWARD

- Utilisée après les couches d'attention, la couche "Feed Forward" permet une **transformation non linéaire** des données.

  - Deux couches pleinement connectées séparées par une fonction d'activation non linéaire.

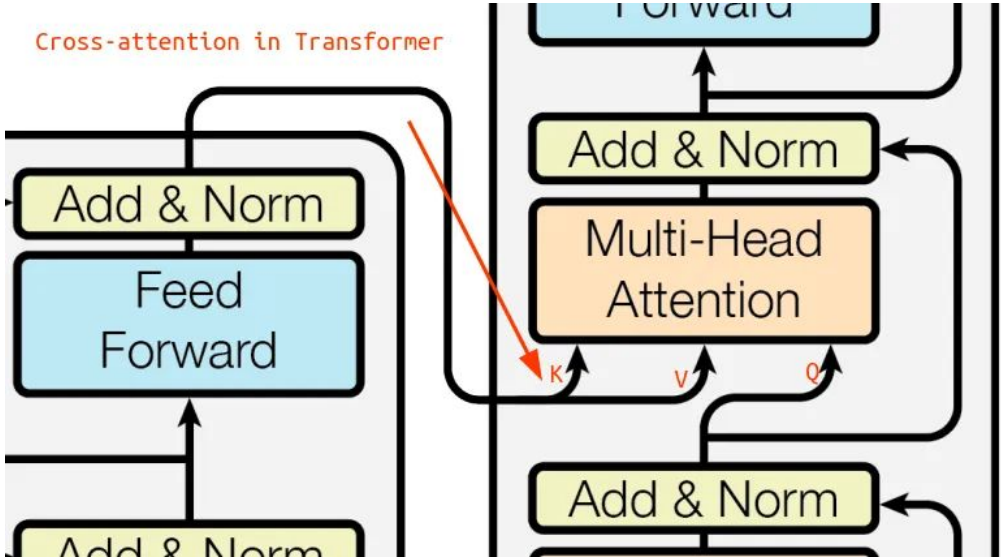
$$FFN(x) = \mathrm{ReLU}(xW_1 + b_1)W_2 + b_2$$

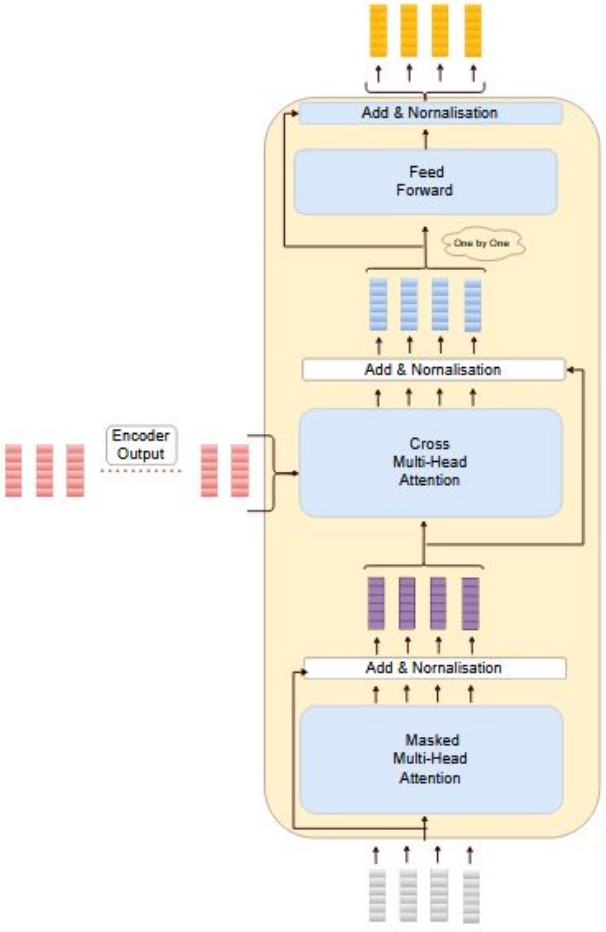
  - W1,W2 : Matrices de poids.
  - b1,b2 : Vecteurs de biais.
  - ReLU (ou GELU, selon les modèles comme GPT ou T5) : Fonction d'activation non linéaire.



La première couche dense projette les données dans un espace de dimension supérieure (souvent 4×d_model). La deuxième couche renvoie les vecteurs dans un espace de dimension d_model.

- Dans les modèles du type SeqToSeq, le mécanisme de **Cross Attention** est similaire à celui vu précédemment cependant :
  - Les matrices **K**, **V** sont calculées avec les sorties de l'encoder.
  - La matrice **Q** donne le nombre de lignes de sortie.





La génération avec le décodeur se fait de manière incrémentale, le nombre de vecteurs d'entrée est toujours égale au nombre de vecteurs de sortie.

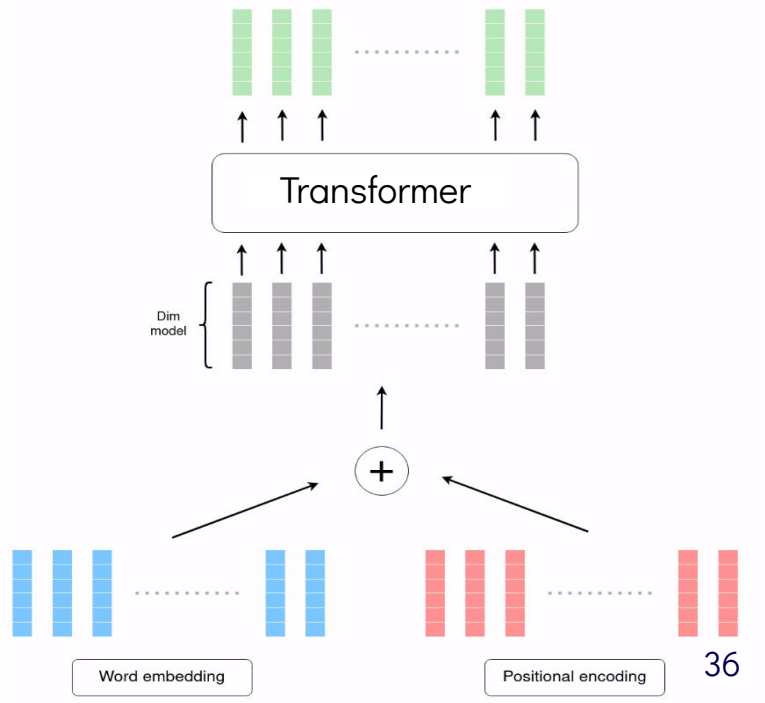
LES POSITIONAL EMBEDDINGS

- Par commutativité de la **somme**, nous remarquons que l'ordre des tokens pourrait être ignoré dans l'architecture d'attention vue précédemment si une information positionnelle n'était pas rajoutée.

  - Garder l'information des positions des tokens dans la phrase en ajoutant à chaque embedding de token un **embedding de position**.

- **Embeddings sinusoidaux :** dans le modèle original de *"Attention is All You Need"*, les auteurs ont proposé une méthode d'encodage positionnel basée sur des fonctions trigonométriques.

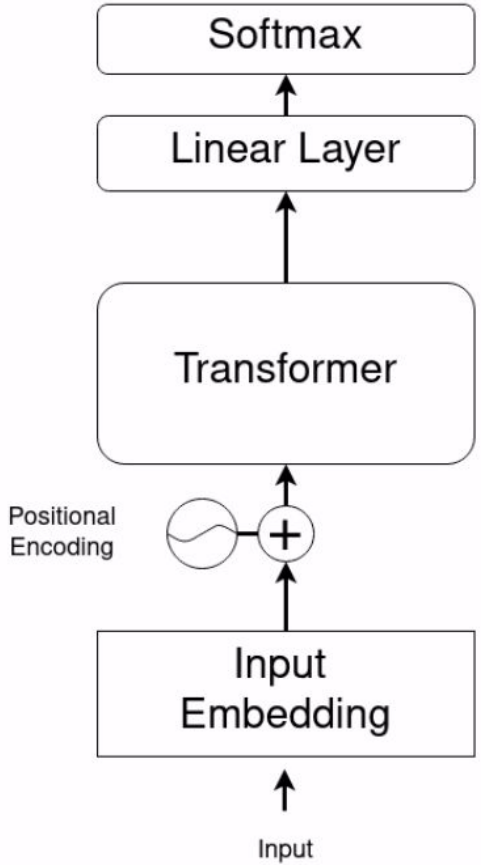
$$PE_{(pos,j)} = \begin{cases} \sin\left(pos/10^{4\times\frac{2\times\lfloor\frac{j}{2}\rfloor}{d_{model}}}\right) & \text{si : } j \equiv 0\ [2] \\ \cos\left(pos/10^{4\times\frac{2\times\lfloor\frac{j}{2}\rfloor}{d_{model}}}\right) & \text{si : } j \equiv 1\ [2] \end{cases}$$



Nous présentons ici l'idée initiale d'ajouter une information positionnelle au modèle, les modèles les plus récents utilisent des embeddings positionnels relatifs au niveau de l'attention.

PRÉ-ENTRAÎNER LES POIDS D'UN MODÈLE

- Apprentissage des poids des modèles basées sur une tâche de **classification multiclasses** :
  - Prédire l'indice d'un token parmi tous les mots du vocabulaire.
  - Produire une mesure de **probabilité conditionnelle** selon un contexte et dépendant des poids du modèle .
- **Auto-supervision** sur les données d'apprentissage avec différentes méthodes :
  - **Maskage** : remplacer aléatoirement les indices de certains tokens via un token **<MASK>** (ex: BERT).
  - **Prédire le prochain token** à la suite d'une séquence de tokens (ex: GPT).
- Ajout d'une **tête de décodage** à la sortie des blocs de Transformers puis passage au **Softmax** sur les logits pour obtenir une mesure de probabilité.



MASKING LANGUAGE MODELING

- Une séquence **X** d'indices de tokens issue des données d'entraînement :

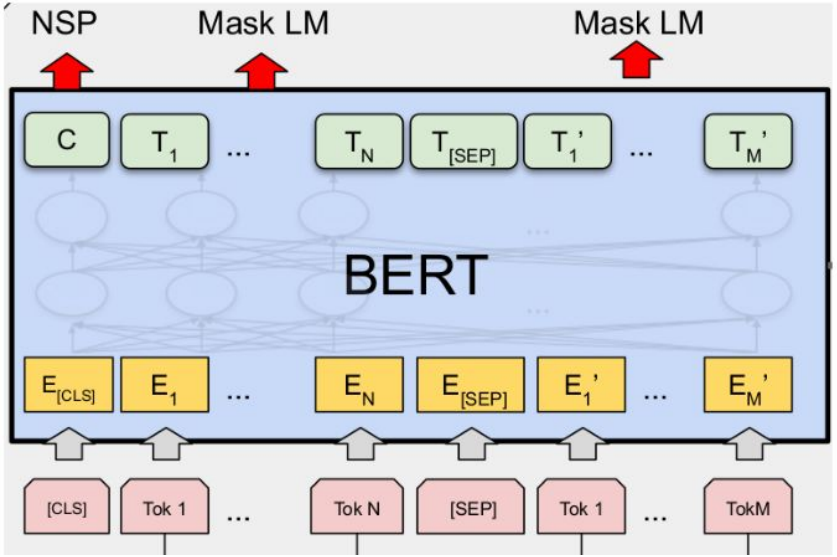
$$X = [x_1, x_2, \ldots, x_n]$$

- **Encoder** type **BERT** (Masked Language Modeling - MLM) est entraîné en remplaçant aléatoirement 15% des indices par :
  - Le token spécial **[MASK]** (80% des cas).
  - Un autre token aléatoire (10% des cas).
  - Restent inchangés (10% des cas).

$$X = [x_1, [\text{MASK}], x_3, \ldots, x_n]$$

- La fonction de perte à minimiser est l'**entropie-croisée** équivalente à la méthode du maximum de vraisemblance.

$$\mathcal{L}_{\text{MLM}} = -\sum_{x_i \in M} \log P(x_i | X_{\text{masked}})$$



Ajout d'un token spécial **<CLS>** utilisé pour entraîner les poids du modèle sur des tâches de classification

NEXT TOKEN PREDICTION

❖ Des couples **(X,Y)** d'indice de tokens issue des données d'entraînement (Ex: question/réponse).

$$X = [x_1, x_2, \ldots, x_n] \qquad Y = [y_1, y_2, \ldots, y_m]$$

Modèle Decoder :

- **Decoder** type **GPT** (Causal Language Modeling - CLM) cherche à maximiser la probabilité du token suivant étant donné tous les tokens précédents.
  - La probabilité de la séquence de sortie complète **P(Y|X)** est factorisée en utilisant la règle de la chaîne

$$P(Y_1, \ldots, Y_m | X) = \prod_{i=1}^{m} P(Y_i | Y_1, \ldots, Y_{i-1}, X)$$

- Minimiser la fonction de perte à minimiser est l'**entropie-croisée** équivalent à la méthode du maximum de vraisemblance.

$$\mathcal{L}_{\text{CLM}} = -\sum_{i=1}^{m} \log P(y_i | y_1, \ldots, y_{i-1}, X)$$

Modèle SeqToSeq :

- **SeqToSeq** type **T5** (Text-to-Text Training) est entraîné à maximiser la probabilité du token suivant en utilisant les sorties de l'encoder une seule fois générés.

$$X = [x_1, x_2, \ldots, x_n] \longrightarrow H_{\text{enc}} = [h_1^{\text{enc}}, h_2^{\text{enc}}, \ldots, h_n^{\text{enc}}]$$

- Les labels Y décalés vers la droite en ajoutant un token spécial de début de séquence **<S>** (<PAD> pour T5)

$$Y' = \text{Input}_{\text{decoder}} = [<\text{s}>, y_1, y_2, \ldots, y_{m-1}]$$

- L'**entropie-croisée** est aussi la fonction de perte à minimiser.

$$\mathcal{L}_{\text{T5}} = -\sum_{t=1}^{m} \log P(y_t | Y'_1, Y'_2, \ldots, Y'_{t-1}, H_{\text{enc}})$$

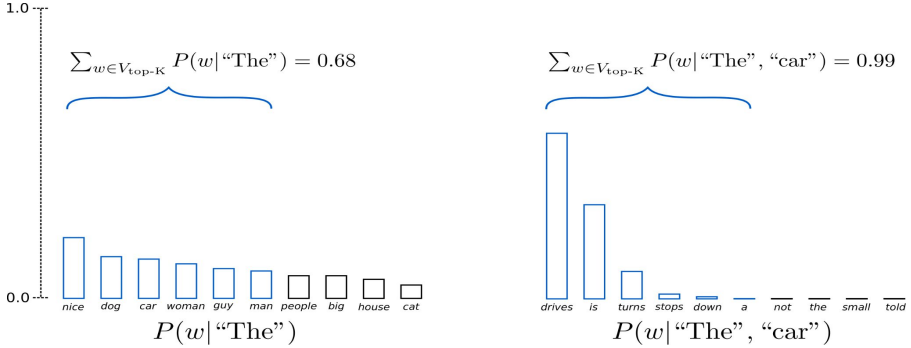
ÉCHANTILLONNER POUR RETROUVER DU TEXTE

Les modèles génératifs produisent des **probabilités conditionnelles** pour chaque mot du vocabulaire, en tenant compte des mots de contexte. **Comment par la générer du contenu textuel ?**

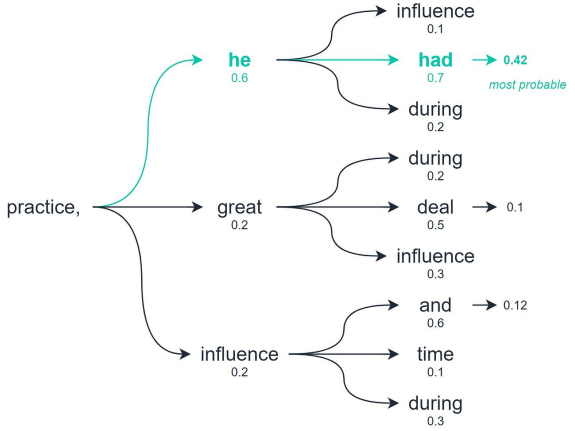
Idée *:* échantillonner “intelligemment” les mots à générer en suivant les lois de probabilité produites par le modèle.

Techniques d’échantillonnage courantes :

- **Greedy Sampling :** correspond à un ***argmax*** sur la mesure de probabilité
- **Nucleus Sampling (Top-p)** : échantillonne parmi les mots dont la probabilité cumulée atteint un seuil **p** (ex : p = 0.9).
- **Top-k Sampling :** restreint l’échantillonnage aux **k** mots les plus probables (ex : k = 5)



- **Beam Search :** explorer plusieurs chemins de génération simultanément et sélectionne le chemin globalement optimal.
- **Sampling sous contrainte :** génère des mots en respectant des contraintes spécifiques, comme des structures grammaticales ou des thèmes imposés. (ex: recette de cuisine)



ÉVALUER LA PERTINENCE DU CONTENU GÉNÉRÉ

- Pour des tâches de **classification non itérative** nous avons l'habitude d'utiliser des métriques de performance humainement compréhensibles telle que :

$$\text{Accuracy} = \frac{\text{Vrai Positifs} + \text{Vrai Négatifs}}{\text{Total des Prédictions}}$$

$$F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

- Ces métriques **ne sont pas adaptées à la génération de séquences** associées à de la classification multiclasse itérative.

Référence :

> L'éducation est cruciale pour réduire les inégalités et promouvoir un développement durable.

Séquence générée :

> Réduire les inégalités et soutenir le développement durable passe par une éducation de qualité.

- Les métriques de classification précédentes évaluent des prédictions avec **exactitude sur l'emplacement des mots**.

- Dans la génération, on s'intéresse plus à des notions de ressemblance, de fluence ou encore de longueure de contenu produit.

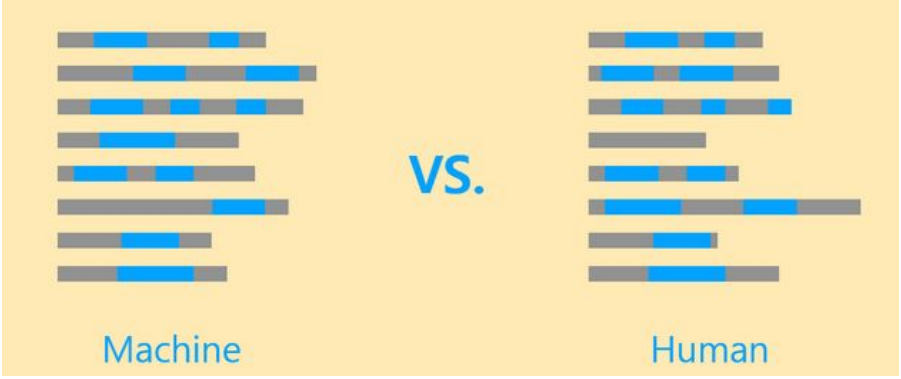
BLEU

- **Précision sur les n-grammes** : BLEU mesure combien de n-grammes du texte généré apparaissent également dans le texte de référence.
- **Pénalité pour répétitions excessives :** un même mot ou segment peut avoir une précision élevée mais sera pénalisée.
- **Pénalité pour les résumés trop courts (Brevity Penalty - BP)**

$$\text{BLEU} = BP \cdot \exp\left(\sum_{n=1}^{N} w_n \cdot \log p_n\right),$$

- N : Taille maximale des n-grammes.
- w_n : Poids pour chaque taille d'ngramme (souvent égal pour toutes les tailles).
- p_n : Précision modifiée des n-grammes de taille n.

Un **n-gramme** est une séquence contiguë de n éléments (mots, caractères, ou autres unités) extraite d'un texte



$$Count_{clip} = \min(Count, Max\_Ref\_Count)$$

$$p_n = \frac{\sum_{C \in \{Candidates\}} \sum_{n\text{-}gram \in C} Count_{clip}(n\text{-}gram)}{\sum_{C' \in \{Candidates\}} \sum_{n\text{-}gram' \in C'} Count(n\text{-}gram')}.$$

$$\text{BP} = \begin{cases} 1 & \text{if } c > r \\ e^{(1-r/c)} & \text{if } c \leq r \end{cases}.$$

ROUGE

- **ROUGE** peut être vu comme une extension du F1 Score à des séquences de texte, mais avec des variantes qui prennent en compte les n-grammes ou des sous-séquences, et non la position exacte des mots.

REF: Today was sunny.

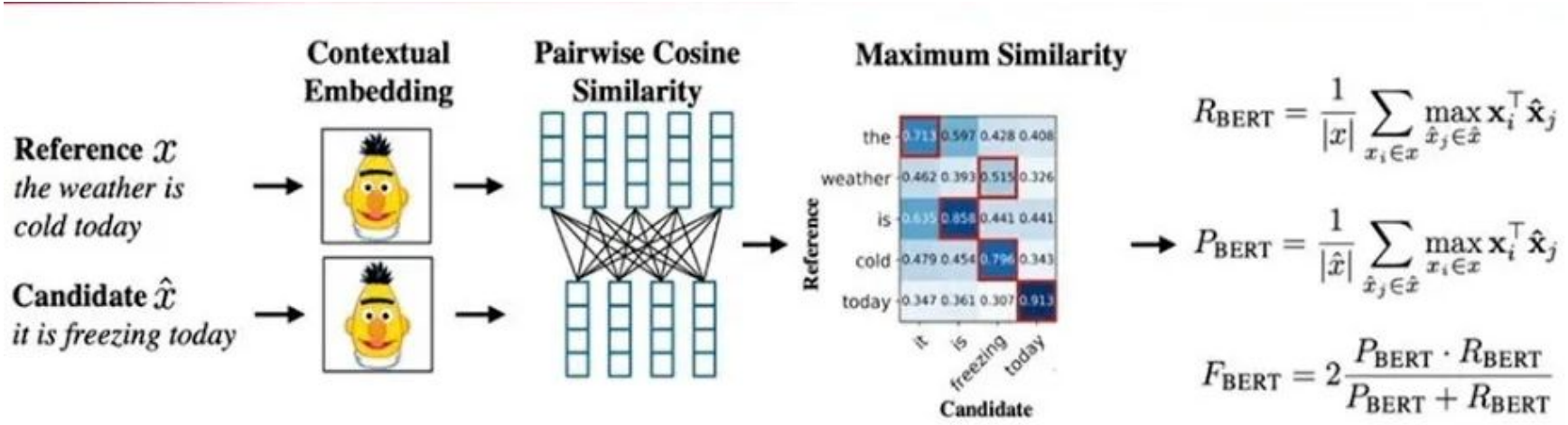
PRED: Today the weather is sunny.

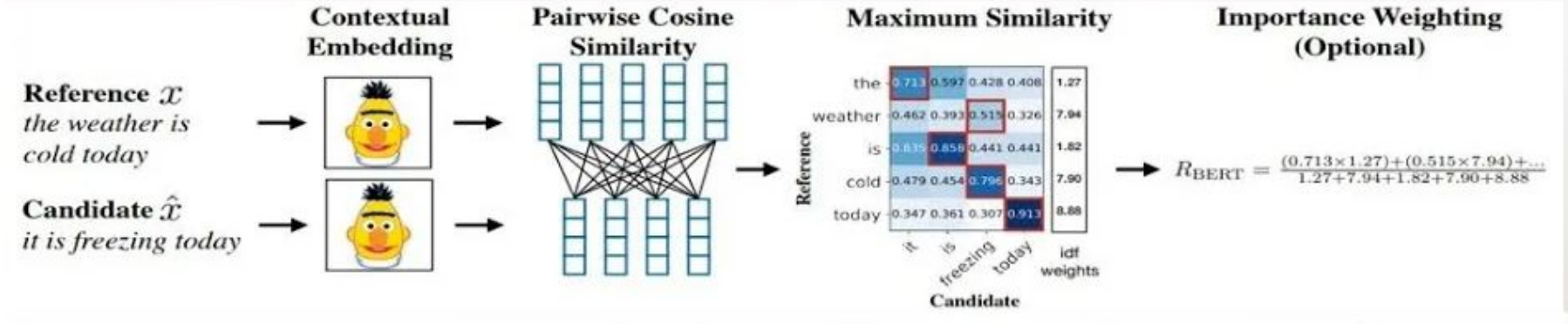
$$\text{Recall} = \frac{\text{n-grammes communs}}{\text{n-grammes totaux dans la référence}}$$

$$\text{Precision} = \frac{\text{n-grammes communs}}{\text{n-grammes totaux générés}}$$

$$\text{ROUGE} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

Introducing **BERTScore**





Ivanhoé BOTCAZOU

THANKS!

CREDITS: This presentation template was created by **Slidesgo** , including icons by **Flaticon** , infographics & images by **Freepik**

Bibliographie :

🏗️ Architectures et Modèles Fondamentaux

- **Word2Vec** MIKOLOV, Tomas, et al. "Efficient estimation of word representations in vector space." *arXiv preprint arXiv:1301.3781*. 2013.
- **Attention Is All You Need (Le Transformer)** VASWANI, Ashish, et al. "Attention is all you need." *Advances in neural information processing systems*. 2017.
- **BERT** DEVLIN, Jacob, et al. "BERT: Pre-training of deep bidirectional transformers for language understanding." *arXiv preprint arXiv:1810.04805*. 2018.
- **GPT-2** RADFORD, Alec, et al. "Language models are unsupervised multitask learners." *OpenAI blog*. 2019.
- **T5 (Text-to-Text Transfer Transformer)** RAFFEL, Colin, et al. "Exploring the limits of transfer learning with a unified text-to-text transformer." *Journal of Machine Learning Research*. 2020.

📏 Métriques d'Évaluation

- **BLEU (Bilingual Evaluation Understudy)** PAPINENI, Kishore, et al. "Bleu: a method for automatic evaluation of machine translation." *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*. 2002.
- **ROUGE (Recall-Oriented Understudy for Gisting Evaluation)** LIN, Chin-Yew. "ROUGE: A package for automatic evaluation of summaries." *Text summarization branches out*. 2004.

TP2 : TITLE GENERATION

Objectif : modifier les poids d'un modèle de langue déjà entraîné en le spécialisant sur la génération de titre pour des articles d'actualité issus d'un ensemble de données *Kaggle*.

SLIDES EN +

Pour des raisons de coût de calcul dans l'implémentation du modèle Word2Vec, l'entraînement ne se fait pas via la prédiction sur tous les mots du vocabulaire par passage au Softmax (le coût de calcul de $\nabla \log p(w \mid w_t)$ est proportionnel à W).

L'entraînement repose sur une classification binaire en échantillonnant aléatoirement un nombre fini de mots hors contexte. Cette approche est appelée **Negative Sampling**.

On définit la probabilité que w soit un mot de contexte de w' par :

$$p(w \mid w') = \sigma(w^\top w') = \frac{1}{1 + \exp(-w^\top w')} \quad \text{(fonction sigmoïde).}$$

$$1 - p(w \mid w') = 1 - \frac{1}{1 + \exp(-w^\top w')} = \sigma(-w^\top w').$$

Pour chaque token central w_t et ses mots de contexte :

$$\mathcal{C} = \{w_{t-c}, \ldots, w_{t-1}, w_{t+1}, \ldots, w_{t+c}\}$$

Un tirage aléatoire de mots hors contexte $\mathcal{N}_\mathcal{C}$ est fait selon une loi $P(w_t)$ qui prends en compte la fréquence des tokens dans le corpus.

La fonction de coût à minimiser en chaque mot central w_t est la suivante :

$$\sum_{w \in \mathcal{C}} -\log \sigma(w^\top w_t) + \sum_{w \in \mathcal{N}_\mathcal{C}} -\log \sigma(-w^\top w_t)$$