Université des Sciences et de la Technologie HOUARI BOUMEDIENE
Faculté d'Informatique
Département d'Intelligence Artificielle et Sciences des Données

Traitement Automatique du Langage Naturel (TALN)

Chapitre II. Mots et Tokens

Séance N° : 1

Ilyes KHENNAK
Maitre de Conférences Classe A

Laboratoire de Recherche en Intelligence Artificielle (LRIA)
Bureau 212, Département IASD, Faculté d'Informatique

Chapitre II. Mots et Tokens

1 Mots

2 Normalisation de Texte

Algorithme de **segmentation**

Algorithme **Byte-Pair Encoding**

3 Distance d'édition

Algorithme **Levenshtein**

Mots

Définition

- Un mot constitue l'unité fondamentale d'un texte, formée par une séquence de caractères.
- En d'autres termes, un texte est constitué d'un ensemble de mots.
- Les mots sont souvent séparés par des espaces.

Exemple

« Sidi Bel Abbes, appelée aussi Bel Abbes ou SBA, est une wilaya algérienne située à l'ouest de l'Algérie. »

Cette phrase comporte 18 mots.

Mots

Définition

- Les espaces ne suffisent pas toujours à distinguer les mots.
- "Sidi Bel Abbes" peut parfois être considéré comme un seul mot.

Exemple

« Sidi Bel Abbes, appelée aussi Bel Abbes ou SBA, est une wilaya algérienne située à l'ouest de l'Algérie. »

Cette phrase comporte 15 mots.

Mots

Définition

- Un mot peut être décomposé en deux parties.
- "l'Algérie" peut être séparé en "La" et "Algérie".

Exemple

« Sidi Bel Abbes, appelée aussi Bel Abbes ou SBA, est une wilaya algérienne située à le ouest de la Algérie. »

Cette phrase comporte 17 mots.

Mots

Définition

- Les signes de ponctuation sont souvent traités comme des mots distincts.

Exemple

« Sidi Bel Abbes , appelée aussi Bel Abbes ou SBA , est une wilaya algérienne située à le ouest de la Algérie . »

Cette phrase comporte 20 mots.

Mots

Définition

- Les **mots** en **majuscules**, comme **"Le"**, et les **mots** en **minuscules**, comme **"le"** :
  - **Parfois, sont considérés comme le identiques.**

    Reconnaissance vocale
  - **Parfois, sont considérés comme des mots différents.**

    Reconnaissance d'entités nommées

Mots

Mots inconnus

- Les **corpus textuels d'apprentissage** se composent de deux types de corpus : **corpus d'entraînement** et **corpus de test**.
  - Les **algorithmes** de **TALN** basés sur l'**apprentissage** utilisent des **corpus textuels d'apprentissage**.
  - Ces **algorithmes apprennent** souvent des **informations linguistiques** à partir d'un **corpus d'entraînement**, qu'ils **utilisent** ensuite pour **prendre** des **décisions** sur un **corpus de test**.
- Les **mots absents** du **corpus d'entraînement** mais **présents** dans le **corpus de test** sont appelés des **mots inconnus**.

Mots

Mots inconnus

Exemple

Si le **corpus d'entraînement** contient les mots **"low"**, **"new"** et **"newer"**, mais pas **"lower"**, et que **"lower"** apparaît dans le **corpus de test**, le système ne saura pas comment le **traiter**

Le mot **"lower"** est considéré comme un **mot inconnu**

Normalisation de texte

Définition

- **Convertir** le texte en une **forme standard**.
- Le **texte** doit être **normalisé avant** tout **traitement** automatique du **langage naturel**.
- La **normalisation** implique généralement **trois tâches** principales :

① **Tokenisation (segmentation) des mots**

② **Normalisation des mots**

③ **Segmentation des phrases**

Normalisation de texte

T①kenisation (segmentation) des mots

- **Segmenter (diviser) un texte en mots, appelés Tokens, en utilisant un algorithme de segmentation (Tokenisation).**

Exemple

Texte : "Deep Learning is playing an essential role in NLP"

Tokens : "Deep", "Learning", "is", "playing", "an", "essential", "role", "in", "NLP"

Normalisation de texte

T①kenisation (segmentation) des mots

- **Algorithme de segmentation**
  - **Traitement d'un mot composé (détaché) comme un seul Token.**

Exemple

« **Sidi Bel Abbes** », « **Bab Ezzouar** », etc.

Normalisation de texte

T①kenisation (segmentation) des mots

- **Algorithme de segmentation**
  - **Traitement des signes de ponctuation comme des Tokens distincts.**

Exemple

Le **point « . »** dans **« est une wilaya algérienne située à le ouest de l'Algérie. »** doit être considéré comme un **Token**, qui indique la **limite de la phrase**.

Normalisation de texte

T①kenisation (segmentation) des mots

- **Algorithme de segmentation**
  - **Maintien de la ponctuation qui se trouve à l'intérieur des mots.**

Exemple

« **U.S.T.H.B.** », « **Ph.D.** », etc.

Normalisation de texte

T①kenisation (segmentation) des mots

- **Algorithme de segmentation**
  - **Maintien de la virgule présente dans les nombres.**

Exemple

« **15,87** », « **0,25** », etc.

Normalisation de texte

T①kenisation (segmentation) des mots

- **Algorithme de segmentation**
  - **Traitement d'un nombre réel comme un seul Token et non comme deux Tokens.**

Exemple

« **15,87** » et non pas « **15** » et « **87** »

Normalisation de texte

T①kenisation (segmentation) des mots

- **Algorithme de segmentation**
  - **Conservation les caractères spéciaux et les chiffres dans les éléments suivants : prix, dates, URLs, hashtags et adresses e-mail.**

Exemple

« **17,78€** », « **01/02/06** », « **https://www.usthb.dz** », « **#USTHB** », « **ikhennak@usthb.dz** », etc.

Normalisation de texte

T①kenisation (segmentation) des mots

- **Exemples d'algorithmes de segmentation**

  ① **Segmentation basée sur les expressions régulières**

  - La fonction de segmentation `nltk.regexp` du **Natural Language Toolkit (NLTK)** basé sur Python permet de **segmenter un texte** en utilisant des **expressions régulières**.

Normalisation de texte

T①kenisation (segmentation) des mots

- **Exemples d'algorithmes de segmentation**

① **Segmentation basée sur les expressions régulières**

Exemple

```
>>> Text ='That U.S.A. poster-print costs $12.40...'
>>> pattern = r''' (?x) # Supprimer commentaires/espaces blancs
...(?:[A-Z]\.)+          # Abréviations, par exemple U.S.A.
...|\w+?:(-\w+)*         # Mots composés
...|\$?\d+(?:\.\d+)?%?   # Devises, Pourcentages
...| \.\.\.              # Points de suspension
...| [][.,;"'?():_`-]
...'''
>>> nltk.regexp_tokenize(Text, pattern)
>>> ['That', 'U.S.A.', 'poster-print', 'costs', '$12.40', '...']
```

Normalisation de texte

T①kenisation (segmentation) des mots

- **Exemples d'algorithmes de segmentation**

  ② **Segmentation basée sur les commandes UNIX**

  - Une version **simple** et **naïve** de la **Tokenisation des mots.**
  - La **commande UNIX** `tr` permet de **segmenter** les **mots** en **remplaçant** chaque **séquence de caractères non alphabétiques** par un **saut de ligne.**

Normalisation de texte

T①kenisation (segmentation) des mots

- **Exemples d'algorithmes de segmentation**

② Segmentation basée sur les commandes UNIX

Exemple

La sortie de la commande :

```
tr -sc 'A-Za-z' '\n' < SBA.txt
```

sera :

```
Sidi
Bel
Abbes
appelée
aussi
Bel
Abbes
est
...
```

Normalisation de texte

T①kenisation (segmentation) des mots

- **Exemples d'algorithmes de segmentation**

  ③ **Segmentation basée sur l'apprentissage**

  - Ces **algorithmes** sont utilisés pour **segmenter** les **mots inconnus** dans des **corpus textuels d'apprentissage**, plus précisément dans les **corpus de test**.
  - Ces **algorithmes génèrent** automatiquement des **ensembles de Tokens**, incluant des **Tokens plus petits** que les **mots inconnus**, appelées **sous-mots connus**.

Normalisation de texte

T①kenisation (segmentation) des mots

- **Exemples d'algorithmes de segmentation**

  ③ **Segmentation basée sur l'apprentissage**

Exemple

Chaque **mot inconnu**, comme **"lower"** peut être représenté par une séquence de **sous-mots connus**, tels que **"low"** et **"er"**

Normalisation de texte

T①kenisation (segmentation) des mots

- **Exemples d'algorithmes de segmentation**

  ③ **Segmentation basée sur l'apprentissage**

  - Les **algorithmes de segmentation** basés sur l'**apprentissage** se **composent** de **deux parties** :

    - **Token learner**

      Prend un **corpus d'entraînement** en entrée et retourne un **vocabulaire**, c'est-à-dire un **ensemble de Tokens**.

    - **Token segmenter**

      Prend un **corpus de test** et le **segmente** en **Tokens** à partir du **vocabulaire**.

Normalisation de texte

Tokenisation (segmentation) des mots

- **Exemples d'algorithmes de segmentation**

  ③ **Segmentation basée sur l'apprentissage**

  - **Les algorithmes de segmentation basés sur l'apprentissage, largement utilisés, sont :**

    ① **WordPiece [2012]**

    ② **Byte-Pair Encoding (BPE) [2016]**

    ③ **Unigram Language Modeling [2018]**

Normalisation de texte

T①kenisation (segmentation) des mots

- **Exemples d'algorithmes de segmentation**
  - Algorithme **Byte-Pair Encoding (BPE)**

1 Token Learner

Normalisation de texte

T①kenisation (segmentation) des mots

- **Exemples d'algorithmes de segmentation**
  - Algorithme **Byte-Pair Encoding (BPE) – Token Learner**
    - **Entrée**

Corpus d'entraînement (C)

Les caractères de chaque mot du corpus sont séparés par des espaces.

Exemple :

C = "newer wider wider new low newer lowest low low newer newer low"

devient

C = "n e w e r w i d e r w i d e r n e w l o w n e w e r l o w e s t l o w l o w n e w e r n e w e r l o w"

Normalisation de texte

T①kenisation (segmentation) des mots

- **Exemples d'algorithmes de segmentation**
  - Algorithme **Byte-Pair Encoding (BPE) – Token Learner**
    - **Entrée**

Corpus d'entraînement (C)

Un symbole spécial est utilisé pour marquer la fin des mots.

Exemple :

C = "n e w e r w i d e r w i d e r n e w l o w n e w e r l o w e s t l o w l o w n e w e r n e w e r l o w"

devient

C = "n e w e r _ _ w i d e r _ _ w i d e r _ _ _ n e w _ _ l o w _ _ n e w e r _ _ l o w e s t _ _ l o w _ _ l o w _ _ n e w e r _ _ n e w e r _ _ l o w _ _"

Normalisation de texte

T①kenisation (segmentation) des mots

- **Exemples d'algorithmes de segmentation**
  - Algorithme **Byte-Pair Encoding (BPE) – Token Learner**
    - **Entrée**

      K

      Nombre de nouveaux Tokens à générer.

      Exemple :

      K = 6

Normalisation de texte

T①kenisation (segmentation) des mots

- **Exemples d'algorithmes de segmentation**
  - Algorithme **Byte-Pair Encoding (BPE) – Token Learner**
    - **Sortie**

      V

      Vocabulaire composé de tous les caractères du corpus d'entraînement **(C)** plus **K** nouveaux Tokens.

Normalisation de texte

T①kenisation (segmentation) des mots

- **Exemples d'algorithmes de segmentation**
  - Algorithme **Byte-Pair Encoding (BPE) – Token Learner**
    - **Actions**

Action n°1 :

Générer le vocabulaire **V** composé de tous les caractères du corpus d'entraînement **(C)**.

Exemple :

C = "n e w e r _ w i d e r _ w i d e r _ _ n e w _ l o w _ n e w e r _ l o w e s t _ l o w _ l o w _ n e w e r _ n e w e r _ l o w _"

V = [d, e, i, l, n, o, r, s, t, w, _]

Normalisation de texte

T①kenisation (segmentation) des mots

- **Exemples d'algorithmes de segmentation**
  - Algorithme **Byte-Pair Encoding (BPE) – Token Learner**
    - **Actions**

Action n°2 :

Choisir les deux **symboles** du vocabulaire **V** les plus fréquemment adjacents dans le corpus d'entraînement **(C)**.

Exemple :

C = "n e w e r _ w i d e r _ w i d e r _ n e w _ l o w _ n e w e r _ l o w e s t _ l o w _ l o w _ n e w e r _ n e w e r _ l o w_"

V = [d, e, i, l, n, o, r, s, t, w, _]

Normalisation de texte

T①kenisation (segmentation) des mots

- **Exemples d'algorithmes de segmentation**
  - Algorithme **Byte-Pair Encoding (BPE) – Token Learner**
    - **Actions**

Action n°3 :

Fusionner les deux symboles choisis et ajouter le symbole généré au vocabulaire **V**.

Exemple :

C = "n e w e r _ _ w i d e r _ _ w i d e r _ _ n e w _ _ l o w _ _ n e w e r _ _ l o w e s t _ _ l o w _ _ l o w _ _ n e w e r _ _ n e w e r _ _ l o w _ _"

V = [d, e, i, l, n, o, r, s, t, w, _ _, er]

Normalisation de texte

T①kenisation (segmentation) des mots

- **Exemples d'algorithmes de segmentation**
  - Algorithme **Byte-Pair Encoding (BPE) – Token Learner**
    - **Actions**

Action n°4 :

Dans le corpus d'entraînement **(C)**, remplacer les deux symboles choisis par le nouveau symbole généré.

Exemple :

C = "n e w er _ _ w i d er _ _ w i d er _ _ n e w _ _ l o w _ _ n e w er _ _ l o w e s t _ _ l o w _ _ l o w _ _ n e w er _ _ n e w er _ _ l o w_ _"

V = [d, e, i, l, n, o, r, s, t, w, _ _, er]

Normalisation de texte

T①kenisation (segmentation) des mots

- **Exemples d'algorithmes de segmentation**
  - Algorithme **Byte-Pair Encoding (BPE) – Token Learner**
    - **Actions**

Action n°5 :

Si le nombre de nouveaux Tokens généré est inferieur à **K**

Alors Aller à l'Action n°1

Sinon Aller à l'Action n°6

Normalisation de texte

T①kenisation (segmentation) des mots

- **Exemples d'algorithmes de segmentation**
  - Algorithme **Byte-Pair Encoding (BPE) – Token Learner**
    - **Actions**

Action n°6 :

Retourner le vocabulaire **V**, composé de tous les caractères du corpus d'entraînement **(C)**, ainsi que des **K** nouveaux Tokens générés.

Exemple :

C = "new er__ w i d er__ w i d er__ new __ low __ new er__ low e s t __ low __ low __ new er__ new er__ low__"

K = 6

V = [d, e, i, l, n, o, r, s, t, w, __, er, re__, ne, new, lo, low]

Normalisation de texte

T①kenisation (segmentation) des mots

- **Exemples d'algorithmes de segmentation**
  - Algorithme **Byte-Pair Encoding (BPE)**

2 Token Segmenter

Normalisation de texte

T①kenisation (segmentation) des mots

- **Exemples d'algorithmes de segmentation**
  - Algorithme **Byte-Pair Encoding (BPE) – Token Segmenter**
    - **Entrée**

Corpus de test (T)

Un symbole spécial est utilisé pour indiquer la fin de mot.

Exemple :

T = "lower new"

devient

T = "l o w e r _ n e w _"

Normalisation de texte

T①kenisation (segmentation) des mots

- **Exemples d'algorithmes de segmentation**
  - Algorithme **Byte-Pair Encoding (BPE) – Token Segmenter**
    - **Entrée**

      V

      Vocabulaire composé de tous les caractères du corpus d'entraînement **(C)** et les **K** Tokens générés par l'algorithme de **Token Learner**.

      Exemple :

      V = [d, e, i, l, n, o, r, s, t, w, __, er, re__, ne, new, lo, low]

Normalisation de texte

T①kenisation (segmentation) des mots

- **Exemples d'algorithmes de segmentation**
  - Algorithme **Byte-Pair Encoding (BPE) – Token Segmenter**
    - **Sortie**

      Corpus de test (T) segmenté

Normalisation de texte

T①kenisation (segmentation) des mots

- **Exemples d'algorithmes de segmentation**
  - Algorithme **Byte-Pair Encoding (BPE) – Token Segmenter**
    - **Actions**

      Segmenter le corpus de test **(T)** à l'aide des Tokens du vocabulaire **V**.

      Exemple :

      V = [d, e, i, l, n, o, r, s, t, w, __, er, re__, ne, new, lo, low]

      T = "l o w e r __ n e w __"

      devient

      T = "low er__ new __"

Normalisation de texte

T①kenisation (segmentation) des mots

- **Exemples d'algorithmes de segmentation**
  - Algorithme **Byte-Pair Encoding (BPE)**

Le mot inconnu "lower" du corpus de test sera donc représenté par une séquence de deux sous-mots connus : "low" et "er_"

Normalisation de texte

N②rmalisation des mots

- **Mettre les Mots/Tokens dans un format standard, en choisissant une forme unique pour les mots ayant plusieurs formes.**

Exemple

Choisir une forme unique pour les mots **"DZ"** et **"DZD"**

Normalisation de texte

N②rmalisation des mots

- **Mettre tous les mots en minuscule est un type de normalisation.**
  - **Cette normalisation peut être aussi utile dans de nombreuses tâches :**

    Recherche d'information, Reconnaissance vocale
  - **En revanche, cette normalisation peut ne pas être utile dans d'autres tâches :**

    Analyse des sentiments, Classification de texte, Traduction automatique

Exemple

Préserver la différence entre « **US** » en tant que nation et « **us** » en tant que pronom peut être utile.

Normalisation de texte

N②rmalisation des mots

- **Lemmatisation**
  - **Déterminer que deux mots partagent la même racine, malgré leurs différences.**

Exemple

« **am** », « **is** » et « **are** » partagent la même racine « **be** »

« **player** » et « **playing** » partagent la même racine « **play** »

Normalisation de texte

N②rmalisation des mots

- **Lemmatisation**
  - Les **méthodes** de **lemmatisation** les plus sophistiquées **impliquent** une **analyse morphologique** du mot.
  - La **morphologie** est l'**étude** de la manière dont les **mots** sont **construits** à partir d'**unités plus petites** appelées **morphèmes**.

Normalisation de texte

N②rmalisation des mots

- **Lemmatisation**
  - **Deux grandes classes de morphèmes peuvent être distinguées :**
    - **Radicaux : le morphème central du mot, fournissant le sens principal.**
    - **Affixes : modifiant le sens principal du mot.**

Exemple

Le mot « **player** » est constitué de deux morphèmes : le morphème « **play** » et le morphème « **er** ». Un **analyseur morphologique** prend un mot comme « **player** » et le décompose en deux morphèmes, « **play** » et « **er** »

Normalisation de texte

N②rmalisation des mots

- **Lemmatisation**
  - Les **algorithmes de lemmatisation** peuvent être complexes.
    - Une méthode plus simple **consiste** à **supprimer** les **affixes** des mots.
    - Cette méthode naïve de l'analyse morphologique est appelée **Stemming**.
    - **Porter Stemmer** est un algorithme de **Stemming** largement utilisé.

Normalisation de texte

S③egmentation des phrases

- **Cette étape consiste à diviser un texte en phrases individuelles en utilisant des indices, comme les signes de ponctuation.**

Exemple

Texte : "This is obvious ! Deep Learning is playing an essential role in NLP."

Phrases : "This is obvious", “Deep Learning is playing an essential role in NLP"

Distance d'édition

Définition

- **Une grande partie du TALN consiste à évaluer le degré de similarité entre deux chaînes de caractères.**

Exemple

Correction des erreurs orthographiques

L'utilisateur a saisi une chaîne incorrecte (par exemple : « **graffe** »). Il est probable que l'utilisateur souhaitait utiliser un mot proche de « **graffe** ». Parmi les mots candidats similaires, on trouve « **girafe** », qui ne diffère de « **graffe** » que par une seule lettre

Distance d'édition

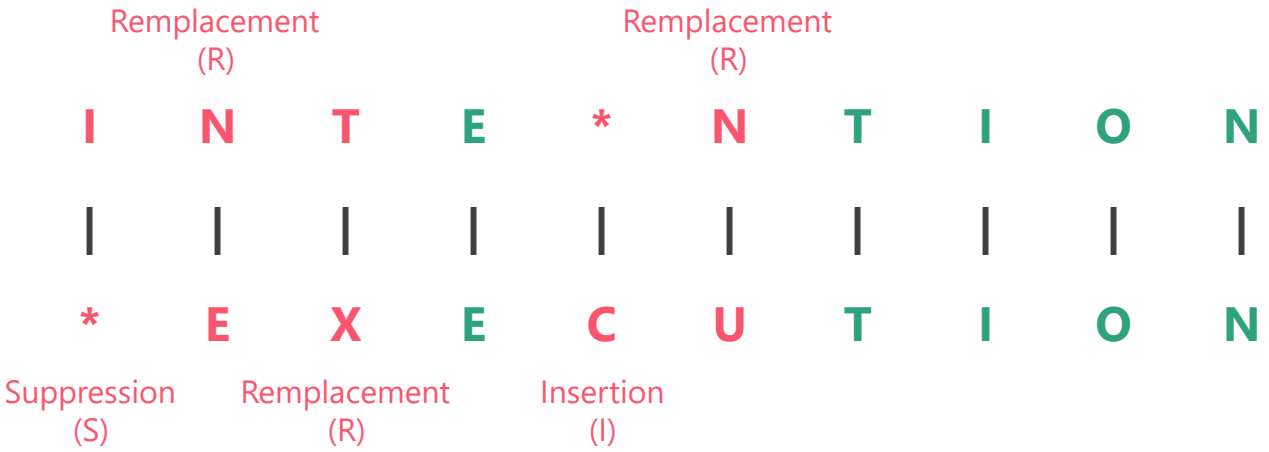
Définition

- La **distance d'édition** permet de **quantifier** la **similarité** entre **deux chaînes de caractères**.
- La **distance d'édition minimale** est le **nombre minimal** d'**opérations d'édition**, telles que l'**insertion**, la **suppression** ou la **substitution**, nécessaires pour **transformer** une chaîne en une autre.

Distance d'édition

Définition

Exemple



L'écart entre « **INTENTION** » et « **EXECUTION** », par exemple, est de **5** (supprimer un **I**, remplacer **E** par **N**, remplacer **X** par **T**, insérer **C**, remplacer **U** par **N**)

Distance d'édition

Distance de Levenshtein

- **Version standard**
  - Attribue un coût à chaque opération d'édition : insertion, suppression ou remplacement.
  - Chaque opération d'édition est assignée un coût de 1.
  - La substitution d'une lettre par elle-même a un coût de 0.
  - La distance de Levenshtein entre les mots "INTENTION" et "EXECUTION" est de 5.

Distance d'édition

Distance de Levenshtein

- **Version alternative**
  - La **substitution** est équivalente à une **insertion** suivie d'une **suppression**.
  - Cela permet d'**autoriser** la **substitution** tout en lui attribuant un **coût** de **2**.
  - La **substitution** d'une lettre par **elle-même** reste à **coût nul**.
  - En utilisant cette version, la **distance de Levenshtein** entre **"INTENTION"** et **"EXECUTION"** est de **8**.

Distance d'édition

Distance de Levenshtein : algorithme

- **Entrée**

  Source (S)

  Mot source

  Exemple :

  S = "INTENTION"

  Cible (T)

  Mot cible

  Exemple :

  T = "EXECUTION"

Distance d'édition

Distance de Levenshtein : algorithme

- **Entrée**

  N

  Taille du mot source

  Exemple :

  N = 9

  M

  Taille du mot cible

  Exemple :

  N = 9

Distance d'édition

Distance de Levenshtein : algorithme

- **Sortie**

  Distance

  Distance d'édition minimale

Distance d'édition

Distance de Levenshtein : algorithme

- **Actions**

  Action n°1 :

  Créer une matrice de distances **D** de taille **(N+1) * (M+1)**

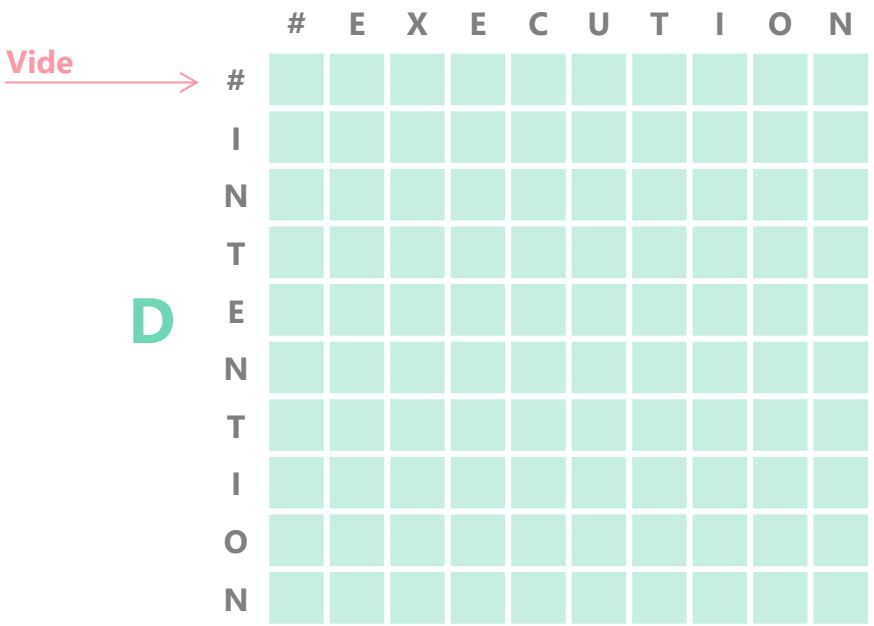
Distance d'édition

Distance de Levenshtein : algorithme

- **Actions**

  Action n°1 :

  Exemple :



Distance d'édition

Distance de Levenshtein : algorithme

- **Actions**

  Action n°2 :

  Initialiser la distance **D[0, 0]** à **0**

Distance d'édition

Distance de Levenshtein : algorithme

- **Actions**

  Action n°2 :

  Exemple :

D

	#	E	X	E	C	U	T	I	O	N
#	0									
I										
N										
T										
E										
N										
T										
I										
O										
N										

Distance d'édition

Distance de Levenshtein : algorithme

- **Actions**

  Action n°3 :

  Pour chaque ligne **i** de **1** à **N** Faire

  Initialiser la distance **D[i, 0]** à **D[i-1, 0]** + **coût_suppression_de_(S[i])**

Distance d'édition

Distance de Levenshtein : algorithme

- **Actions**

  Action n°3 :

  Exemple :

coût_suppression_de_(S[i]) = 1

D

	#	E	X	E	C	U	T	I	O	N
#	0									
I	1									
N	2									
T	3									
E	4									
N	5									
T	6									
I	7									
O	8									
N	9									

Distance d'édition

Distance de Levenshtein : algorithme

- **Actions**

  Action n°4 :

  Pour chaque colonne **j** de **1** à **M** Faire

  Initialiser la distance **D[0, j]** à **D[0, j]** + **coût_insertion_de_(T[j])**

Distance d'édition

Distance de Levenshtein : algorithme

- **Actions**

  Action n°4 :

  Exemple :

coût_insertion_de_(T[j]) = 1

D

	#	E	X	E	C	U	T	I	O	N
#	0	1	2	3	4	5	6	7	8	9
I	1									
N	2									
T	3									
E	4									
N	5									
T	6									
I	7									
O	8									
N	9									

Distance d'édition

Distance de Levenshtein : algorithme

- **Actions**

  Action n°5 :

  Pour chaque ligne **i** de **1** à **N** Faire

  Pour chaque colonne **j** de **1** à **M** Faire

  Initialiser la distance **D[i, j]** à :

  Min (**D[i-1, j]** + **coût_supression_de_(S[i])**,

  D[i, j-1] + **coût_insertion_de_(T[j])**,

  D[i-1, j-1] + **coût_substitution_de_(S[i], T[j])** **)**

Distance d'édition

Distance de Levenshtein : algorithme

- **Actions**

  Action n°5 :

  Exemple :

coût_suppression_de_(S[i]) = 1

coût_insertion_de_(T[j]) = 1

coût_substitution_de_(S[i], T[j]) = 0
(S[i] = T[j])

coût_substitution_de_(S[i], T[j]) = 2
(S[i] <> T[j])

D

	#	E	X	E	C	U	T	I	O	N
#	0	1	2	3	4	5	6	7	8	9
I	1	2	3	4	5	6	7	6	7	8
N	2	3	4	5	6	7	8	7	8	7
T	3	4	5	6	7	8	7	8	9	8
E	4	3	4	5	6	7	8	9	10	9
N	5	4	5	6	7	8	9	10	11	10
T	6	5	6	7	8	9	8	9	10	11
I	7	6	7	8	9	10	9	8	9	10
O	8	7	8	9	10	11	10	9	8	9
N	9	8	9	10	10	12	11	10	9	8

Distance d'édition

Distance de Levenshtein : algorithme

- **Actions**

  Action n°6 :

  Initialiser Distance à **D[N, M]**

Distance d'édition

Distance de Levenshtein : algorithme

- **Actions**

  Action n°6 :

  Exemple :

D

		#	E	X	E	C	U	T	I	O	N
	#	0	1	2	3	4	5	6	7	8	9
	I	1	2	3	4	5	6	7	6	7	8
	N	2	3	4	5	6	7	8	7	8	7
	T	3	4	5	6	7	8	7	8	9	8
	E	4	3	4	5	6	7	8	9	10	9
	N	5	4	5	6	7	8	9	10	11	10
	T	6	5	6	7	8	9	8	9	10	11
	I	7	6	7	8	9	10	9	8	9	10
	O	8	7	8	9	10	11	10	9	8	9
	N	9	8	9	10	10	12	11	10	9	8

Distance = 8

Distance d'édition

Distance de Levenshtein : algorithme

- **Actions**

  Action n°7 :

  Retourner **Distance**

Distance d'édition

Alignement

- L'**alignement** de deux chaînes est **essentiel** en **TALN**.
  - En **reconnaissance vocale** ou en **traduction automatique**, l'**alignement** est utilisé pour calculer des métriques comme le **Taux d'Erreur sur les Mots (WER)**.

Exemple

I	N	T	E	*	N	T	I	O	N	
|	|	|	|	|	|	|	|	|	|	Alignement
*	E	X	E	C	U	T	I	O	N	

Distance d'édition

Alignement

- **Un chemin à travers la matrice de distance d'édition.**
  - **Fournir une visualisation ou une représentation de la distance minimale d'édition entre deux chaînes.**

Exemple

Les **cellules** en **rouge** représentent un **alignement**.

Deux cellules en **rouge** dans la **même ligne** indiquent une **insertion**.

Deux cellules en **rouge** dans la **même Colonne** indiquent une **suppression**.

	#	E	X	E	C	U	T	I	O	N
#	0	1	2	3	4	5	6	7	8	9
I	**1**	2	3	4	5	6	7	6	7	8
N	2	**3**	4	5	6	7	8	7	8	7
T	3	4	**5**	6	7	8	7	8	9	8
E	4	3	4	**5**	**6**	7	8	9	10	9
N	5	4	5	6	7	**8**	9	10	11	10
T	6	5	6	7	8	9	**8**	9	10	11
I	7	6	7	8	9	10	9	**8**	9	10
O	8	7	8	9	10	11	10	9	**8**	9
N	9	8	9	10	10	12	11	10	9	**8**

Distance d'édition

Alignement

- Le **calcul** du chemin d'**alignement** se fait en **deux étapes**.

1ère Etape

- **Mettre à jour** l'algorithme de la **distance d'édition minimale** pour **enregistrer** des **pointeurs de retour "BACKPOINTERS"** dans chaque cellule.
- Le **BACKPOINTER** d'une cellule indique la ou les **cellules précédentes** par lesquelles on est **arrivé** à la **cellule actuelle**.
- Certaines **cellules** peuvent avoir **plusieurs BACKPOINTERS**, car la **solution optimale** peut provenir de **plusieurs cellules** antérieures.

Distance d'édition

Alignement

Exemple

D

	#	E	X	E	C	U	T	I	O	N
#	0	← 1	← 2	← 3	← 4	← 5	← 6	← 7	← 8	← 9
I	↑ **1**	↖ ↑ ← 2	↖ ↑ ← 3	↖ ↑ ← 4	↖ ↑ ← 5	↖ ↑ ← 6	↖ ↑ ← 7	↖ 6	← 7	← 8
N	↑ 2	↖ ↑ ← **3**	↖ ↑ ← 4	↖ ↑ ← 5	↖ ↑ ← 6	↖ ↑ ← 7	↖ ↑ ← 8	↑ 7	↖ ↑ ← 8	↖ 7
T	↑ 3	↖ ↑ ← 4	↖ ↑ ← **5**	↖ ↑ ← 6	↖ ↑ ← 7	↖ ↑ ← 8	↖ 7	↑ ← 8	↖ ↑ ← 9	↑ 8
E	↑ 4	↖ 3	← 4	↖ ← **5**	← **6**	← 7	↑ ← 8	↖ ↑ ← 9	↖ ↑ ← 10	↑ 9
N	↑ 5	↑ 4	↖ ↑ ← 5	↖ ↑ ← 6	↖ ↑ ← 7	↖ ↑ ← **8**	↖ ↑ ← 9	↖ ↑ ← 10	↖ ↑ ← 11	↖ ↑ 10
T	↑ 6	↑ 5	↖ ↑ ← 6	↖ ↑ ← 7	↖ ↑ ← 8	↖ ↑ ← 9	↖ **8**	← 9	← 10	↑ ← 11
I	↑ 7	↑ 6	↖ ↑ ← 7	↖ ↑ ← 8	↖ ↑ ← 9	↖ ↑ ← 10	↑ 9	↖ **8**	← 9	← 10
O	↑ 8	↑ 7	↖ ↑ ← 8	↖ ↑ ← 9	↖ ↑ ← 10	↖ ↑ ← 11	↑ 10	↑ 9	↖ **8**	← 9
N	↑ 9	↑ 8	↖ ↑ ← 9	↖ ↑ ← 10	↖ ↑ ← 10	↖ ↑ ← 12	↑ 11	↑ 10	↑ 9	↖ **8**

Distance d'édition

Alignement

- Le **calcul** du chemin d'**alignement** se fait en **deux étapes**.

2ème Etape

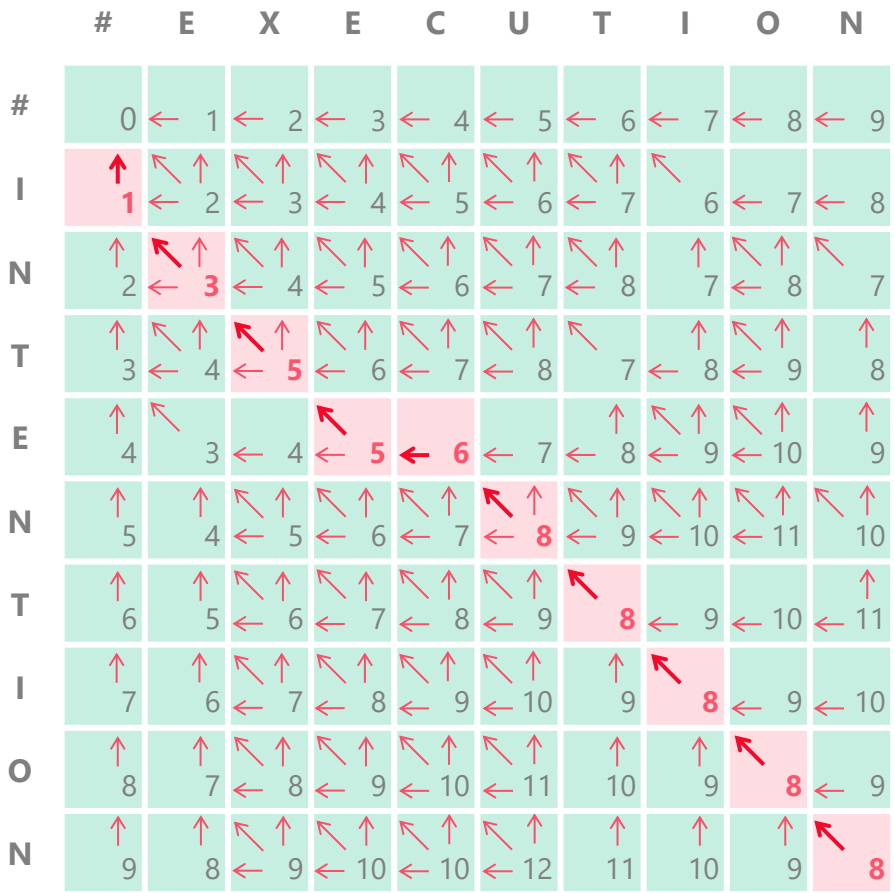
- Effectuer un **BACKTRACE**. Cette étape consiste à partir de la **dernière cellule** et à **remonter** en suivant les **BACKPOINTERS** à travers la **matrice de distance**.
- Chaque **chemin complet**, de la **cellule finale** à la cellule de départ, correspond à un **alignement de distance minimale**.

Distance d'édition

Alignement

D

Exemple



Fin de la séance N° 1