



DEVOIR 2  
Hiver 2015

Mouna Mokaddem

Souhila Benbetka

IFT6255

## Introduction :

Un système de recherche d'information performant doit être capable de satisfaire le besoin d'information de l'utilisateur qui l'exprime à partir d'une requête donnée, et cela dépend de deux phases nécessaires : la phase d'indexation et la phase de recherche d'information.

La disponibilité d'un nombre volumineux de données (ie documents) et le nombre limité de mots clés au niveau de la requête ainsi que la différence au niveau du vocabulaire des requêtes et des documents, toutes ces conditions rendent la tâche du système de recherche d'information plus difficile et crée beaucoup de bruit au niveau des documents retrouvés. L'utilisation de stoplist et de stemming améliore les résultats mais ne résout pas le problème. L'expansion de la requête est une technique qui permet d'étendre la requête par l'ajout d'un ensemble de mots qui sont potentiellement liés sémantiquement avec les termes de la requête initiale permettant de récupérer les documents les plus pertinents. Il existe trois approches d'expansion de requête: une approche statistique qui se base sur la corrélation entre les mots basée sur leurs co-occurrence. Elle présente deux type; l'approche globale qui se base sur les documents indexés et l'approche locale qui utilise les documents résultants d'une première requête. La deuxième classe d'approche utilise les logs de requêtes. Finalement, la troisième classe utilise des ressources linguistiques (cette classe n'est pas couverte pas notre travail).

Dans ce travail, nous essayerons de voir l'impact de l'utilisation de l'expansion de la requête sur les résultats du système de recherche d'information Indri. Nous utiliserons la collection de test TREC. Nous commencerons par l'application d'une approche locale avec utilisation de documents de feedback et nous expérimenterons par la suite une approche globale qui utilise la relation de co-occurrence entre les termes aussi. Après nous essayerons de faire l'expansion avec anchor log. Et en définitive nous discuterons les différents résultats.

## Résumé de nos expérimentations :

Nous résumons dans cette partie les observations que nous avons pu tirer des expérimentations. Et dans la suite du rapport, nous montrerons en détail les résultats et l'analyse de ces résultats que nous avons eu pour chaque modèle des trois modèles sur lequel nous avons travaillé.

Nous remarquons, en premier lieu, que l'expansion de requête s'avère utile pour des ensembles de requêtes et non utile pour d'autres. Quand elle est utile elle augmente le rappel et/ou la précision. En effet, cette technique permet d'étendre la requête par l'ajout d'un ensemble de mots qui sont fortement liés ce qui permet d'enrichir le vocabulaire de la requête et d'élargir le champ de recherche permettant ainsi de récupérer plus de documents pertinents. Cependant pour certains ensembles de requêtes, l'expansion de requêtes n'est pas utile car elle augmente le risque de "query drift" c'est à dire le risque de déviation par rapport à l'intention initiale de l'utilisateur en ajoutant des mots qui n'ont pas de dépendances sémantiques avec les mots de la requête initiale augmentant ainsi

le bruit dans les documents récupérés. Il faut signaler aussi que l'expansion de requêtes, bien qu'elle est utile, elle augmente le temps de recherche d'information. Par conséquent, il faut trouver un compromis entre temps de recherche et qualité des résultats de la recherche.

Nous avons exploré trois différentes techniques pour l'expansion de la requête. Nous allons citer, dans ce qui suit, un résumé de nos observations pour chacune d'entre elles :

### **Expansion de requête avec les documents de feedback :**

Nous avons remarqué que globalement les résultats obtenus avec expansion de requêtes sont meilleurs que ceux obtenus avec les requêtes initiales car la décision initiale du système est renforcée en rendant la requête étendue plus semblable aux documents pertinents récupérés lors de la première itération. Nous avons constaté aussi que l'augmentation (ou diminution dans quelques cas) du taux de rappel et/ou précision dépend de deux paramètres qui sont le nombre de documents et de termes de feedback.



### **Expansion de requête en utilisant la relation de co-occurrence entre les termes :**

Nous avons constaté que cette technique permet d'améliorer le taux de rappel et/ou précision à l'exception de quelques cas où nous avons enregistré une détérioration par rapport aux taux enregistrés par les requêtes sans expansion. Ceci peut s'expliquer par le choix du paramètre fenêtre de co-occurrence car les termes trouvés à une distance de  $-9$  ou  $+9$  ne sont pas obligatoirement reliés sémantiquement avec les termes de la requête initiale. Par conséquent, il faut faire varier cette fenêtre pour trouver de meilleurs résultats. Cette variation n'est pas effectuée dans notre travail mais peut en être une extension potentielle.

### **Expansion de requête en utilisant l'anchor log :**

Les résultats enregistrés pour cette technique n'ont pas été tranchants. En effet, ça n'a pas été clair s'il y a une amélioration des taux de rappel et/ou précision par rapport à la requête initiale. De plus, nous n'avons pas remarqué que cette technique favorise un ensemble de requêtes et pas les autres. Nous penchons à croire que cette technique n'améliore pas la performance par rapport à la requête initiale et ceci parce que les ressources utilisées pour l'expansion sont extérieures à la collection. Par conséquent, même si les mêmes mots sont utilisés pour les requêtes, le contexte varie largement.

Il faut noter la présence de deux paramètres supplémentaires qui influencent la performance des techniques qui sont les suivants :

-  La valeur du poids qui est de 0.5 dans ce travail. Nous pouvons la faire varier pour améliorer les résultats.
-  Pour les deux dernières techniques, le nombre total de termes ajouté à chaque requête dépend du nombre total de termes de la requête initiale. En effet, une requête plus longue qu'une autre va être favorisée par la technique. Et ceci diminuera la performance d'une technique. Par conséquent, il faut penser (dans une extension potentielle de ce travail) à normaliser les nouvelles requêtes.

## Partie 1 : Expansion de requête avec les documents de feedback

La première approche d'expansion à tester est l'expansion de requête avec pseudo-relevance feedback. Dans ce devoir, nous utilisons le modèle de pertinence (relevance model) proposé par Lavrenko qui est déjà intégré dans indri. En effet dans le fichier paramètre de la commande IndriRunQuery, il faut ajouter trois champs qui représentent les paramètres de pseudo-relevance feedback qui sont les suivants :

`<fbDocs>` : un entier spécifiant le nombre de documents de feedback à utiliser.

`<fbTerms>` : un entier spécifiant le nombre de termes de feedback à utiliser.

`<fbOrigweight>` : un nombre en virgule flottante inclus dans l'intervalle [0.0 .. 1.0] spécifiant le poids de la requête initiale dans la requête étendue.

La *figure 1* montre le nouveau fichier de paramètres de la commande IndriRunQuery après l'ajout des paramètres que nous avons susmentionnés.

```
<query>
<number>44</number>
<text>Staff Reductions at Computers and Communications Companies</text>
</query>
<query>
<number>45</number>
<text>What Makes CASE succeed or fail</text>
</query>
<query>
<number>46</number>
<text>Tracking Computer Virus Outbreaks</text>
</query>
<query>
<number>47</number>
<text>Contracts for Computer systems in excess of 1 Million</text>
</query>
<query>
<number>48</number>
<text>Purchasers of Modern Communications Equipment</text>
</query>
<query>
<number>49</number>
<text>Who s working with Supercomputers</text>
</query>
<query>
<number>50</number>
<text>Potential Military Interest in Virtual Reality Applications</text>
</query>
<rule>method:dirichlet,mu=2500</rule>
<fbDocs>20</fbDocs>
<fbTerms>20</fbTerms>
<fbOrigweight>0.5</fbOrigweight>
<runID>runName</runID>
<trecFormat>true</trecFormat>
</parameters>
```

Figure 1: Fichier paramètre de la commande IndriRunQuery avec la méthode pseudo-relevance feedback de Lavrenko

Comme demandé dans l'énoncé du devoir, nous avons fait nos expérimentations avec le modèle de langue ( $\mu = 2000$  comme pour le devoir 1) avec stemming Porter et stoplist et nous allons essayer de comparer les valeurs que nous avons obtenues du devoir 1 avec les nouvelles valeurs obtenues en

faisant l'expansion de requête avec les documents de feedback.

### Résultats partie 1:

Dans nos résultats nous avons étudié les résultats des variables (MAP, P@10, Rappel) que nous procure la partie d'évaluation avec la commande TREC-EVAL.

**Tableau 1:** Valeur de MAP pour les différents ensemble de requêtes avec variation de nombre de documents et de termes de feedback (Modèle de langue avec stemming Porter et stoplist)

MAP	10 docs/10 terms	10 docs/15 terms	10 docs/20 terms	20 docs/10 terms	20 docs/15 terms	20 docs/20 terms
Topic 1-50	0.1454	0.1501	0.1528	0.1455	0.1638	0.1653
Topic 51-100	0.2756	0.2750	0.2747	0.2730	0.2713	0.2707
Topic 101-150	0.2103	0.2118	0.2138	0.2086	0.2151	0.2171

### Observation :

Pour les ensembles de requêtes 1-50 et 101-150, nous remarquons qu'avec l'ajout de 20 documents et de 20 termes à la requête initiale, nous obtenons la valeur la plus élevée du MAP; 0.1653 et 0.2171 respectivement. Par contre pour l'ensemble de requêtes 51-100, la valeur la plus élevée du MAP est de 0.2756 trouvée avec l'ajout de 10 documents et 10 termes à la requête initiale.

**Tableau 2 :** Valeur de MAP pour les requêtes sans expansions et les requêtes avec avec et sans expansion (Modèle de langue avec stemming Porter et stoplist)

MAP	Requête sans expansion	Requête avec expansion
Topic 1-50	0.1554	0.1653
Topic 51-100	0.2262	0.2756
Topic 101-150	0.1630	0.2171

### Observation :

Nous remarquons en voyant le *tableau2* qu'avec l'ajout de documents de feedback à la requête initiale, la valeur du MAP augmente considérablement; 0.1653 contre 0.1554 pour le premier ensemble de requêtes, 0.2756 contre 0.2262 pour le deuxième ensemble de requêtes et 0.2171 contre 0.1630 pour le troisième ensemble de requêtes. Cependant pour le 1er ensemble de requête, nous observons d'après la *tableau1* que l'augmentation de la valeur du MAP n'est pas réalisée pour n'importe quelle valeur d'ajout des documents de feedback et de termes car pour l'ajout de 10 documents avec 10, 15 et 20 termes et l'ajout de 20 documents avec 10 termes, nous remarquons une baisse de la valeur de MAP (par exemple 0.1455 contre 0.1554). Par contre pour l'ajout de 20 documents de feedback avec 15 et 20 termes respectivement, nous observons une augmentation de la valeur du MAP (0.1638 et 0.1653 respectivement contre 0.1554).

Nous remarquons aussi que pour le deuxième et troisième ensemble de requêtes, la valeur du MAP est augmentée quel que soit le nombre de documents et de termes de feedback ajoutés.

**Tableau 3:** Valeur de P@10 pour les différents ensemble de requêtes avec variation de nombre de documents et de termes de feedback (Modèle de langue avec stemming Porter et stoplist)

P@10	10 docs/10 terms	10 docs/15 terms	10 docs/20 terms	20 docs/10 terms	20 docs/15 terms	20 docs/20 terms
Topic 1-50	0.2060	0.1940	0.1940	0.1940	0.1940	0.2000
Topic 51-100	0.4560	0.4540	0.4660	0.4480	0.4580	0.4520
Topic 101-150	0.3480	0.3640	0.3780	0.3540	0.3620	0.3740

### Observation :

Pour les requêtes 1-50, la valeur la plus élevée du P@10 (la précision pour les 10 premiers documents) est de 0.2060 trouvée avec l'ajout de 10 documents et 10 termes à la requête initiale. Par contre pour les ensembles de requêtes 51-100 et 101-150, nous remarquons qu'avec l'ajout de 10 documents et de 20 termes à la requête initiale, nous obtenons la valeur la plus élevée du P@10; 0.4660 et 0.3780 respectivement.

**Tableau 4:** Valeur de P@10 pour les différents ensemble de requêtes avec et sans expansion (Modèle de langue avec stemming Porter et stoplist)

P@10	Requête sans expansion	Requête avec expansion
Topic 1-50	0.1760	0.2060
Topic 51-100	0.4220	0.4660
Topic 101-150	0.3300	0.3780

### Observation :

Le *tableau 4* nous indique qu'avec l'ajout de documents de feedback à la requête initiale, la valeur du P@10 augmente considérablement; 0.2060 contre 0.1760 pour le premier ensemble de requêtes, 0.4660 contre 0.4220 pour le deuxième ensemble de requêtes et 0.3780 contre 0.3300 pour le troisième ensemble de requêtes. Pour les 3 ensembles de requêtes, nous observons d'après le *tableau 3*, que l'augmentation de la valeur du P@10 est réalisée pour la valeur d'ajout dès la première variation des documents de feedback et des termes au stade de 10 documents et 10 termes. Elle est de 0.1940 contre 0.1760 pour 10 documents et 10 termes feedback et de même pour 20 documents et 10 termes toujours une augmentation de 0.1940 contre 0.1760 pour le premier ensemble de requête par exemple.

**Variable Rappel :** Nous avons voulu tester principalement cette variable car elle représente le ratio calculer par la fraction :  $\text{num\_rel\_ret} / \text{num\_rel}$ . Ce ratio nous permet de vérifier si le rappel est modifié ou pas avec ou sans expansion de requête.

**Tableau 5:** Valeur de Rappel pour les différents ensemble de requêtes avec variation de nombre de documents et de termes de feedback (Modèle de langue avec stemming Porter et stoplist)

Rappel	10 docs/10 terms	10 docs/15 terms	10 docs/20 terms	20 docs/10 terms	20 docs/15 terms	20 docs/20 terms
Topic 1-50	0.4395	0.4424	0.4381	0.4381	0.4451	0.4397
Topic 51-100	0.5107	0.5089	0.5087	0.5051	0.5046	0.5038
Topic 101-150	0.4843	0.4934	0.4964	0.4889	0.5001	0.5094

### Observation :

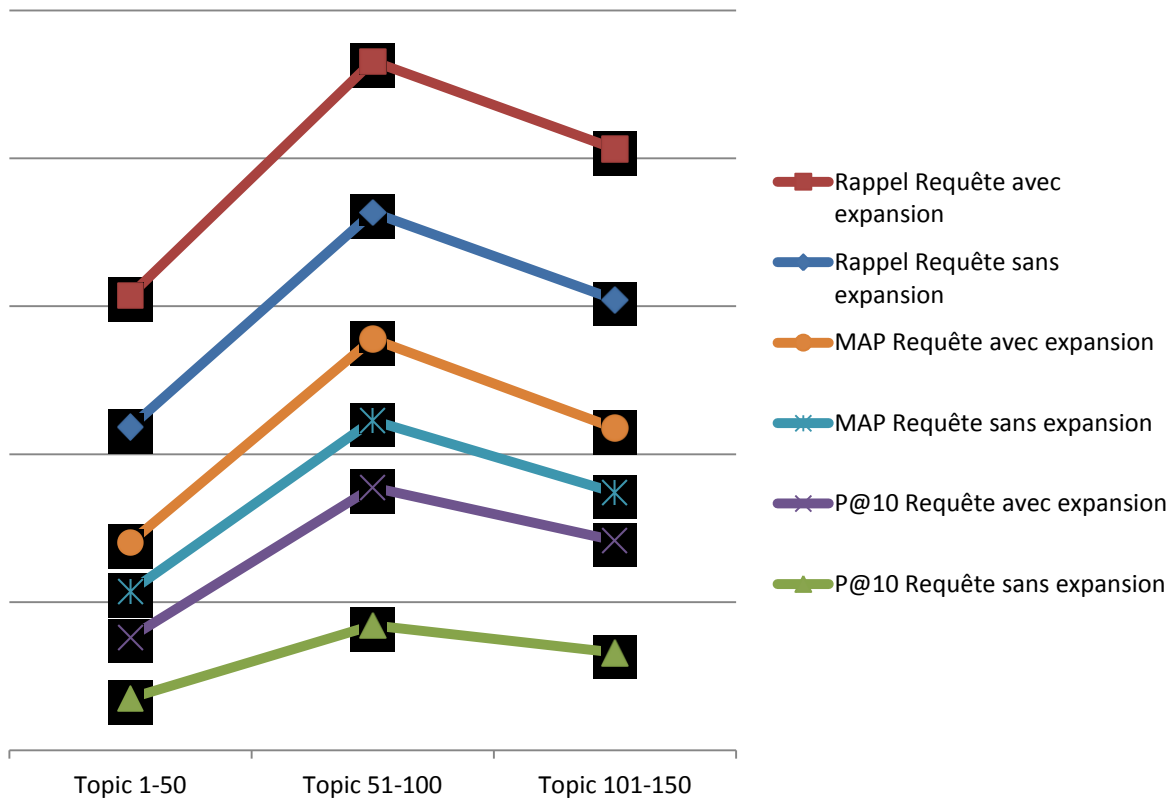
Pour les requêtes 1-50, la valeur la plus élevée du rappel est de 0.4451 trouvée avec l'ajout de 20 documents et 15 termes à la requête initiale. Par contre pour les ensembles de requêtes 51-100 est élevée avec l'ajout de 10 documents et 10 termes à la requête initiale et 101-150, nous remarquons qu'avec l'ajout de 20 documents et de 20 termes à la requête initiale, nous obtenons la valeur la plus élevée du rappel, 0.5107 et 0.5094 respectivement.

**Tableau 6:** Valeur de Rappel pour les différents ensemble de requêtes avec et sans expansion (Modèle de langue avec stemming Porter et stoplist)

Rappel	Requête sans expansion	Requête avec expansion
Topic 1-50	0.3897	0.4451
Topic 51-100	0.4273	0.5107
Topic 101-150	0.4343	0.5094

### Observation :

Le *tableau 6* nous indique qu'avec l'ajout de documents de feedback à la requête initiale, la valeur du rappel augmente considérablement; 0.4451 contre 0.3897 pour le premier ensemble de requêtes, 0.5107 contre 0.4273 pour le deuxième ensemble de requêtes et 0.5097 contre 0.4343 pour le troisième ensemble de requêtes. Pour les 3 ensembles de requêtes, nous observons d'après le *tableau 5*, que l'augmentation de la valeur du rappel est réalisée pour la valeur d'ajout dès la première variation des documents de feedback et des termes au stade de 10 documents et 10 termes. Elle est de 0.4395 contre 0.3897 pour 10 documents et 10 termes feedback et de même pour 20 documents et 10 termes toujours une augmentation de 0.4381 contre 0.3897 pour le premier ensemble de requête par exemple.



**Figure 2: Résumé des variables testées avec et sans expansion pour la méthode feedback**

### Conclusion partie 1:

Nous constatons d'après les courbes de la figure 2 que les résultats obtenus avec expansion de requêtes sont meilleurs que les résultats obtenus avec les requêtes initiales sans expansion. La moyenne de précision (**MAP**), la précision pour les 10 premiers documents (**P@10**) ainsi que le **rappel**, sont constamment améliorés et l'augmentation est de l'ordre de 1%, 3%, 5,5% respectivement pour le premier ensemble de requête, 5%, 4%, 8% respectivement pour le deuxième et de 5,5%, 5%, 7,5% respectivement pour le troisième ensemble. En effet, la pseudo-relevance feedback renforce essentiellement la décision initiale du système en rendant la requête étendue plus semblable aux documents pertinents récupérés lors de la recherche antécédente. Nous pouvons alors conclure que l'ajout de la bonne valeur de documents de feedback et de termes permet d'améliorer la performance de la recherche et la plupart du temps donne de bons résultats. Cependant la recherche prend plus de temps que la recherche initiale étant donné qu'elle tient en compte des résultats des documents récupérés dans la recherche antécédente.



## Partie 2 : Expansion de requête en utilisant la relation de co-occurrence entre les termes

### Description du modèle :

Nous avons implanté notre propre méthode qui nous permet de faire l'extraction des relations entre les termes en utilisant une analyse de co-occurrence sur la collection AP. Nous supposons que deux termes qui co-occurrent dans la collection ont une certaine relation entre eux. Dans ce travail nous limitons la distance de co-occurrence à 10 mots et ensuite nous ne gardons que les mots qui ont une fréquence de co-occurrence d'au moins 10. Par la suite nous utilisons ces résultats pour pondérer la relation entre deux termes en utilisant la mesure suivante.

$$P(t_j | t_i) = \frac{\#cooc(t_i, t_j)}{\sum_{t_k} \#cooc(t_i, t_k)}$$

Après, nous utilisons cette mesure de pondération pour intégrer la relation de co-occurrence dans le modèle initial de la requête avec l'utilisation de la mesure suivante:

$$P(t | \theta_Q) = (1 - \lambda)P_{ML}(t | \theta_Q) + \lambda \sum_{t_i \in Q} P(t | t_i)P_{ML}(t_i | \theta_Q)$$

Nous allons avoir comme résultats une nouvelle requête qui combine les termes initiaux de la requête et les nouveaux termes qui se caractérisent par des nouveaux poids qui représentent le degré de liaison avec les termes de la requête initiale. La dernière étape consiste à réaliser la phase de recherche d'information avec les nouvelles requêtes étendue pour les 3 ensembles de requêtes.

### Implémentation :

La phase d'indexation nous donne comme résultats des indexes des documents de la collection. Nous avons choisi l'index avec le stemming Porter et stoplist comme l'énoncé l'a indiqué. Ensuite, en utilisant la commande Indri ***dumpindex*** avec l'option ***dv*** qui fournit l'ensemble des vecteurs des documents c'est à dire l'ensemble de mots indexés d'un document donné de la collection. Nous avons récupéré tous les vecteurs de tous les documents dans un fichier que nous appelons ***documentvector***. Chaque ligne dans ce fichier correspond à un vecteur d'un document. Le programme que nous avons implémenté prend le fichier ***documentvector*** et le fichier des requêtes prétraitées (en appliquant le stemming Porter et la stoplist) comme input et fait appel aux méthodes suivantes :

### 1) Calculer le nombre de co-occurrence

Nous avons implémenté notre programme avec le langage Java. Nous avons créé une classe **Term** qui représente un terme  $t_i$  de la requête où sont stockés toutes les caractéristiques du terme en question comme le numéro de la requête à laquelle le terme appartient, le terme en question, le poids du terme, la liste des termes qui co-occurrent avec lui, ..etc. La méthode **readQueryFile** se charge de faire le parsing du fichier des requêtes et de calculer le poids de chaque terme et donne en output un tableau de termes d'un ensemble donné de requêtes. Les termes  $t_j$  qui co-occurrent avec un terme  $t_i$  de la requête sont stockés dans une structure de données qui correspond à une hashmap où la clé est le terme  $t_j$  et la valeur est sa fréquence de co-occurrence.

Etant donné un terme  $t_i$  d'une requête donnée avec une position  $p_i$ , l'idée de notre algorithme consiste à parcourir le **documentvector**, une fois le terme trouvé nous récupérons les termes  $t_j$  qui sont dans l'intervalle  $[p_i-9, p_i+9]$  et vérifions si le terme existe, nous ajoutons de +1 sa fréquence d'occurrence sinon nous le stockons dans la hashmap comme un nouveau terme potentiel. Ce travail est assuré par la méthode **countCoocc**. Il faut noter que notre méthode prend en considération dans le calcul toutes les occurrences (apparition) d'un terme  $t_i$  dans un même vecteur d'un document  $d$ , elle ne se limite pas à la première apparition du terme  $t_i$  mais regarde pour toutes les apparitions de ce terme là dans le vecteur du document  $d$  et ceci est fait pour chaque vecteur de document du fichier **documentvector**. Ensuite, La méthode **computeMoreTen** sélectionne les termes  $t_j$  qui sont avec une occurrence d'au moins 10 et retourne un tableau de termes où chaque terme  $t_i$  a comme attribut une hashmap où stockés les termes qui co-occurrent avec lui avec une fréquence d'au moins 10.

### 2) Calculer la somme des co-occurrences

L'étape suivante est assurée par la méthode **TermCoocAllTerms** qui calcule pour un terme  $t_i$  la somme des occurrences des termes qui co-occurrent avec lui et la stocke dans un attribut de la classe **Term** pour l'utiliser après dans le calcul des probabilité  $P(t_j|t_i)$ .

### 3) Calculer $P(t_j|t_i)$

Par la suite, le calcul de probabilité de chaque terme est effectué par la méthode **calculAndSortProbability** et les dix meilleures probabilités sont sélectionnées par la méthode **TenHighestProbabilities**. La méthode **calculAndSortProbability** récupère pour un terme  $t_j$ , qui co-occure avec un terme  $t_i$ , son occurrence puis la divise sur la somme des co-occurrence calculée par la méthode **TermCoocAllTerms** et donne en output un tableau de termes  $t_j$  qui ont comme attribut

supplémentaire un tableau de termes  $t_j$  et de leurs probabilité correspondantes.

#### 4) Calculer le poids des termes $t_j$ qui co-occurrent avec un terme $t_i$

L'étape finale est assurée par la méthode *CalculWeightCoocTerms* qui calcule la formule suivante :  $\sum P(t|t_i)Pml(t_i|\theta_q)$

En effet pour un terme faire pour chaque terme  $t_j$  les étapes suivantes :

- ✚ Si le terme  $t_j$  co-occure seulement avec le terme  $t_i$  alors son poids serait le produit du poids de  $t_i$  avec la probabilité  $P(t_j/t_i)$
- ✚ Sinon c'est la somme du produit calculé précédemment avec une autre quantité qui est calculée comme suit : parcourir le tableau des termes  $t_i$  et voir si le terme  $t_j$  co-occure avec le terme  $t_i$  en cours alors calculer une quantité comme celle susmentionnée (le produit du poids de  $t_i$  avec la probabilité  $P(t_j/t_i)$ ) et ainsi de suite.

#### Résultats du programme :

Notre programme génère un fichier où chaque ligne est sous le format suivant :

Numéro de la requête – Terme de la requête – le poids du terme – L'ensemble de termes qui co-occurrent avec lui avec une fréquence d'au moins 10 (au plus 10) avec leurs poids correspondants

La *figure 2* montre un exemple d'un fichier résultat de notre programme pour l'expansion de requêtes en utilisant la relation de co-occurrence entre terme qui correspond à l'ensemble de requêtes de 1-50. Et la figure 3 montre le nouveau fichier de paramètres de la commande IndriRunQuery après l'ajout des termes de co-occurrence avec leurs poids correspondants.

```
27 neural 0.1667 {}
30 os2 0.5 {}
45 fail 0.25 {two=0.0017, new=0.0148, million=0.0025, govern=0.0034, presid=0.0131, state=0.0056, feder=0.003, report=0.0015, sai=0.0031, nation=0.0024}
34 isdn 0.1111 {}
11 program 0.5 {million=0.0041, new=0.0084, govern=0.0027, percent=0.0023, plan=0.0022, program=0.007, billion=0.0025, state=0.0035, feder=0.0022, nation=0.0023}
23 agrochem 0.3333 {}
4 debt 0.5 {million=0.0091, new=0.005, l=0.0038, percent=0.0053, govern=0.0049, debt=0.0094, compani=0.006, billion=0.0108, bank=0.0065, countri=0.0047}
44 reduct 0.2 {budget=0.0025, new=0.0104, percent=0.0041, tax=0.0028, soviet=0.0031, billion=0.0042, state=0.0059, bush=0.0044, cut=0.0049, reduct=0.0023}
29 att 0.2 {}
34 entiti 0.1111 {govern=0.0877, plan=0.0606}
28 effort 0.3333 {unit=0.002, new=0.0104, govern=0.0026, presid=0.0026, soviet=0.0016, state=0.0096, report=0.0017, bush=0.0019, offici=0.0067, nation=0.0021}
38 impact 0.25 {new=0.0115, percent=0.0034, price=0.0031, market=0.0031, govern=0.0094, state=0.0136, offici=0.004, sai=0.0093, industri=0.0032, nation=0.0058}
34 build 0.1111 {peopl=0.0014, new=0.0062, million=0.0014, polic=7.0E-4, state=0.0398, report=0.0019, offici=0.0032, build=0.0014, citi=8.0E-4, fire=8.0E-4}
32 outsourc 0.3333 {}
4 reschedul 0.5 {meet=0.5}
38 law 0.25 {law=0.0033, peopl=0.0066, new=0.0115, govern=0.0094, state=0.0136, feder=0.0013, court=0.0076, sai=0.0093, offici=0.004, nation=0.0058}
18 stock 0.25 {point=0.0036, share=0.0035, new=0.0131, stock=0.0214, market=0.017, unchang=0.0027, percent=0.0165, trade=0.0136, american=0.0075, volum=0.0057}
6 third 0.25 {two=0.002, 2=0.0017, million=0.0139, new=0.0111, l=0.0038, percent=0.0062, second=0.0018, billion=0.007, state=0.0082, report=0.0018}
19 protectionist 0.5 {trade=0.5}
25 chernobyl 0.5 {plant=0.0528, peopl=0.0528, nuclear=0.0775, april=0.0458, chernobyl=0.0423, soviet=0.1162, 31=0.0352, radiat=0.0387, power=0.0387}
44 compani 0.2 {million=0.0035, new=0.0104, share=0.0013, l=0.0011, percent=0.0041, stock=0.001, compani=0.0056, billion=0.0042, busi=0.0023, corp=0.0025}
20 lawsuit 0.3333 {lawsuit=0.0037, million=0.0041, new=0.031, judg=0.0037, compani=0.1044, file=0.0041, state=0.0054, feder=0.0626, court=0.1296, attornei=0.0039}
42 commit 0.3333 {law=0.0104, system=0.0130, bush=0.0034, billion=0.0054, market=0.0030, compani=0.0044, percent=0.0030, busi=0.0023, attornei=0.0031, commit=0.0037}
```

**Figure 3 : Fichier output de notre programme pour le Topic 1-50 avec l'expansion de la requête en utilisant la relation de co-occurrence entre termes**

```

<parameters>
<index>c:\devoirlift6255\index\index5_output</index>
<count>1000</count>
<query>
<number>1</number>
<text>#weight(0.5 #weight(0.3333 antitrust 0.3333 case 0.3333 pending) 0.5 #weight(
0.024 antitrust 0.0187 law 0.0341 new 0.02 justic 0.016 price 0.024 gener 0.0387
compani 0.0298 feder 0.0281 state 0.0539 court 0.0341 new 0.0073 judg 0.0018 percent
0.0085 charg 0.0068 rule 0.0281 state 0.0298 feder 0.0539 court 0.0024 attorney
0.0055 case 0.0048 two 0.0341 new 0.005 million 0.0073 judg 0.0085 charg 0.0068 rule
0.0281 state 0.0063 appeal 0.0539 court 0.0047 offici))
</text>
</query>
<query>
<number>2</number>
<text>#weight(0.5 #weight(1.0 acquisit) 0.5 #weight(0.0285 million 0.0233 new 0.018
co 0.0151 time 0.0512 compani 0.0244 billion 0.0163 busi 0.0186 bank 0.0244 corp
0.0157 sale)) </text>
</query>
<query>
<number>3</number>
<text> #weight(0.5 #weight(0.5 joint 0.5 ventur) 0.5 #weight(0.0234 new 0.01 million
0.0079 announc 0.011 ventur 0.0076 percent 0.0275 compani 0.0194 soviet 0.015 busi
0.0174 state 0.0084 corp 0.0234 new 0.01 million 0.0079 announc 0.011 ventur 0.0076
percent 0.0275 compani 0.0194 soviet 0.015 busi 0.0174 state 0.0084 corp)) </text>
</query>
<query>
.
.
.
</query>
<rule>method:dirichlet,mu=2000</rule>
<!--<baseline>okapi</baseline> -->
<count>1000</count>
<runID>runName</runID>
<trecFormat>true</trecFormat>
</parameters>

```

**Figure 4 : Fichier paramètre de la commande IndriRunQuery**

## Résultats partie 2:

Nous avons construit un ensemble de requêtes qui combine les termes de la requête initiale avec les nouveaux termes. La dernière étape de notre travail est d'exécuter cet ensemble de requêtes en utilisant *IndriRunQuery*. Nous avons utilisé le modèle de langue avec le smoothing Dirichlet ( $\mu=2000$ ) et avec le paramètre  $\lambda=0.5$  pour retrouver les documents pertinents. Et puis pour l'évaluation, nous avons exécuté la commande `trec_eval` sur les fichiers générés par la commande *InriRunQuery*. Nous rapportons dans des tables ce qui suit les résultats de nos expérimentations avec cette technique d'expansion.

**Tableau 7:** Valeur de MAP pour les différents ensemble de requêtes avec la co-occurrence des termes (Modèle de langue avec stemming Porter et stoplist) et  $\lambda=0.5$

MAP	Requête sans expansion	Requête avec expansion
Topic 1-50	0.1554	0.1279
Topic 51-100	0.2262	0.2210
Topic 101-150	0.1630	0.2092

### Observation :

Pour les deux premiers ensembles de requête 1-50 et 51-100, la valeur de MAP a baissé en utilisant l'expansion de requêtes : 0.1279, 0.2210 contre 0.1554, 0.2260 respectivement. Par contre pour l'ensemble de requêtes 51-100, on remarque une augmentation remarquable de la valeur de MAP 0.2092 contre 0.1630, en utilisant l'expansion de requêtes.

**Tableau 8:** Valeur de P@10 pour les différents ensemble de requêtes avec la co-occurrence des termes (Modèle de langue avec stemming Porter et stoplist) et  $\lambda=0.5$

P@10	Requête sans expansion	Requête avec expansion
Topic 1-50	0.1760	0.1900
Topic 51-100	0.4220	0.4200
Topic 101-150	0.3300	0.4060

### Observation :

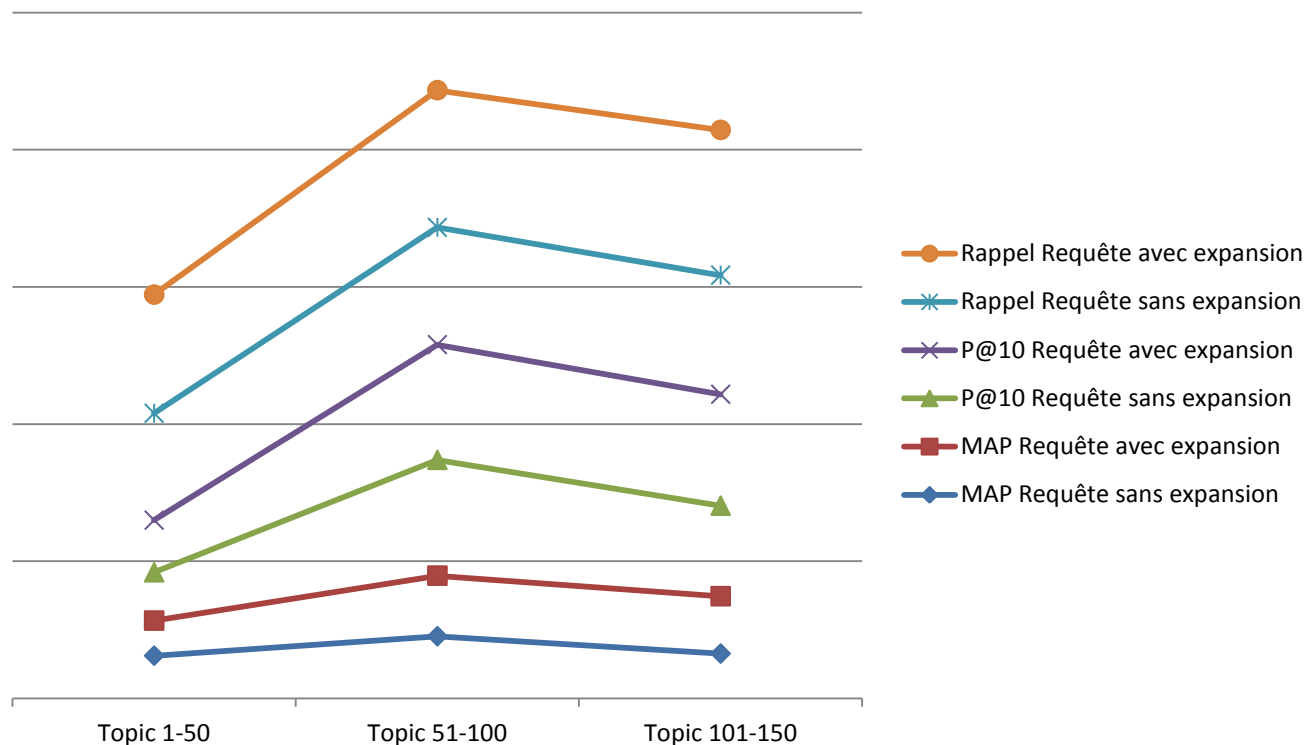
Pour la variable P@10 l'expansion des requêtes était très bénéfique car pour l'ensemble des Topics pour les dix premiers documents la valeur a augmenté 0.1760, 0.4220, 0.3300 contre 0.1900, 0.4200, 0.4060. Malgré qu'on remarque une fine diminution pour le 2em Topics cela n'empêche pas de déduire une amélioration de P@10.

**Tableau 9:** Valeur du Rappel pour les différents ensemble de requêtes avec la co-occurrence des termes (Modèle de langue avec stemming Porter et stoplist) et  $\lambda=0.5$

Rappel	Requête sans expansion	Requête avec expansion
Topic 1-50	0.3897	0.4325
Topic 51-100	0.4273	0.5002
Topic 101-150	0.4343	0.5294

### Observation :

Pour l'ensemble des Topics 1-50, 51-100, 101-150, la valeur du rappel a augmenté en utilisant l'expansion de requête. On a des valeurs de : 0.3897 contre 0.4325 pour les requêtes 1-50, 0.4273 contre 0.5002 pour les requêtes 51-100, 0.4343 contre 0.5294 pour les requêtes 101-150.



**Figure 5: Résumé des variables testées avec et sans expansion pour la méthode de co-occurrence**

### Conclusion partie 2:

Pour la précision moyenne MAP nous ne remarquons une amélioration qu'au niveau de Topic 3. Cela peut être expliqué par la taille de la fenêtre. Nous avons observé, dans le fichier résultat de notre programme, des termes d'extension comme "000" qui est la suite d'un nombre de 6 chiffre par exemple le nombre « 800000 » écrit dans les documents sous la forme « 800 000 » ou un terme de la requête comme "antitrust" qui a un terme d'expansion qui est exactement le même c'est à dire "antitrust". C'est clair que ces termes là n'ajoute rien à la sémantique de la requête étendue car pour le premier cas le terme « 000 » n'a aucune dépendance avec n'importe quel terme et pour le deuxième cas le terme "antitrust" représente une répétition du terme de la requête. Cela est dû bien entendu à la taille de la fenêtre qui nous donne des termes comme ceux susmentionnés qui ajoutent que du bruit aux documents récupérés.

Pour  $P@10$  nous remarquons l'amélioration au niveau du Topic 1 et le Topics 3 et une légère diminution pour le Topic 2.

C'est le rappel qui tranche nous constatons une fluctuation de la valeur du ratio :

Rappel=  $\text{num\_rel\_ret} / \text{num\_rel}$  pour l'ensemble des Topic.

### Partie 3 : Expansion de requête en utilisant l'anchor log

#### Description du modèle :

Le fichier Anchor log est un fichier qui présente un ensemble de paires (anchor, titre) où anchor est le texte d'anchrage d'un lien hypertexte et titre est le titre du document auquel le lien hypertexte envoie. Dans ce travail, nous utilisons un ensemble de paires extrait des articles de Wikipédia.

La relation de co-occurrence de deux termes  $t_i$  et  $t_j$  est définie de la façon suivante : on dit qu'un terme  $t_j$  co-occure avec un terme  $t_i$  du titre si on trouve le terme  $t_j$  dans l'anchor correspond au titre où se trouve le terme  $t_i$  c'est à dire sur la même ligne dans le fichier. Et les fréquences sont toujours considérées égales à 1. Et donc pour calculer les co-occurrences, il faut voir pour chaque terme du titre les termes qui co-occurrent avec lui dans toutes les lignes, les récupérer ainsi que la fréquence de chacun.

Nous remarquons que seule le calcul de la probabilité  $P(t_j | t_i)$  qui change ici par rapport à la deuxième partie. Par conséquent, nous allons dans la partie suivante décrire seulement les nouvelles méthodes pour le calcul de co-occurrence. Le reste des méthodes comme celle qui calcule le poids d'un terme ou les poids des termes qui co-occurrent avec lui, restent inchangées donc nous allons utiliser celles de la partie 2.

#### Implémentation :

L'idée de notre programme est de parcourir le fichier Anchor log précisément parcourir sa deuxième colonne, ligne par ligne. Un terme  $t_i$  rencontré, il faut aller vérifier si ce terme est présent dans l'ensemble de requêtes sur lequel nous travaillons. Si non nous passons au terme suivant. Si oui nous ajoutons tous les termes de la colonne 1 (de la même ligne), s'ils n'existent pas, dans la table des termes qui co-occurrent avec le terme en question ou ajoutons +1 à leurs fréquences de co-occurrence, s'ils existent déjà. Ce travail est assuré par la méthode *parseAnchorLogFile* qui retourne un tableau de termes des requêtes qui ont comme attribut une hashmap de termes qui co-occurrent avec eux avec leurs fréquences correspondantes. Une fois ce travail est fait nous pouvons passer le tableau de termes des requêtes aux autres méthodes mentionnées dans la partie 2 une à une pour calculer les poids à la fin.

## Résultats du programme :

Notre programme génère un fichier qui a le même format que le fichier de la partie 2. La *figure 6* montre un exemple d'un fichier résultat de notre programme pour l'expansion de requêtes en utilisant la l'Anchor log qui correspond à l'ensemble de requêtes de 1-50. Et la figure 7 montre le nouveau fichier de paramètres de la commande IndriRunQuery après l'ajout des termes de co-occurrence avec leurs poids correspondants.

Il faut noter que le fichier Anchor log est sans stemming ni stoplist. Donc nous avons travaillé avec les requêtes originales c'est à dire sans prétraitement. Par contre dans le fichier paramètres de la commande IndriRunQuery, nous avons inclu la stoplist (chose que nous n'avons pas faite pour la partie 2 étant donné que les requêtes ont été prétraitées dès le début).

```
1 antitrust 0.3333 {law=0.0671, antitrust=0.0639, unite=0.0778, justice=0.016, division=0.016, sherman=0.0044, department=0.016, state=0.0778, act=0.008, competition=0.0296}
1 cases 0.3333 {}
1 pending 0.3333 {unite=0.0778, resignation=0.0036, congress=0.0287, pending=0.0466, state=0.0778, intellectual=0.0287, property=0.0287, list=0.0287, legislate=0.0287, patent=0.0179}
2 acquisitions 1.0 {}
3 joint 0.5 {unite=0.0168, joint=0.122, staff=0.0053, congress=0.0138, committee=0.0142, company=0.0049, state=0.0164, venture=0.0088, force=0.0076, chief=0.0051}
3 ventures 0.5 {}
4 debt 0.5 {unite=0.0231, sovereign=0.0098, govern=0.02, collateral=0.0148, debt=0.1613, obligate=0.0153, external=0.0092, state=0.0231, crisis=0.0281, public=0.0189}
4 rescheduling 0.5 {}
5 dumping 0.5 {}
5 charges 0.5 {}
6 third 0.25 {republic=0.0034, scotland=0.0029, division=0.0286, football=0.0242, south=0.0045, list=0.0064, france=0.003, north=0.0042, league=0.0249, third=0.0451}
6 world 0.25 {cup=0.0142, junior=0.0028, skate=0.0028, ii=0.0053, list=0.0064, fifa=0.0075, world=0.059, championship=0.0208, war=0.0095, athletic=0.004}
6 debt 0.25 {unite=0.0116, sovereign=0.0049, govern=0.01, collateral=0.0074, debt=0.0806, obligate=0.0076, external=0.0046, state=0.0116, crisis=0.014, public=0.0094}
6 relief 0.25 {terrain=0.0028, humanitarian=0.006, catholic=0.0029, siege=0.0069, emergency=0.003, aid=0.0039, bas=0.0028, act=0.0038, pitch=0.0259, relief=0.0746}
7 us 0.3333 {}
7 budget 0.3333 {unite=0.0403, office=0.0148, budget=0.1489, federal=0.0131, govern=0.0561, committee=0.0129, kingdom=0.005, state=0.0365, manage=0.0153, house=0.0076}
7 deficit 0.3333 {unite=0.0403, budget=0.1489, balance=0.007, govern=0.0561, deficit=0.0832, disorder=0.0085, debt=0.0075, trade=0.0068, state=0.0365, public=0.0068}
8 economic 0.5 {sector=0.0584, unite=0.0078, nobel=0.0044, recession=0.0068, primary=0.058, economy=0.0702, state=0.005, crisis=0.0048, council=0.0055, economic=0.0746}
8 projections 0.5 {}
```

**Figure 6: Fichier output de notre programme pour le Topic 1-50 avec l'expansion de la requête en utilisant l'Anchor log**

```
<query>
<number>1</number>
<text>#weight(0.5 #weight(0.3333 antitrust 0.3333 cases 0.3333 pending) 0.5 #weight (0.0671 law 0.0639 antitrust 0.0778 unite 0.016 justice 0.016 division 0.0044 sherman 0.016 department 0.0778 state 0.008 act 0.0296 competition 0.0778 unite 0.0036 resignation 0.0287 congress 0.0466 pending 0.0778 state 0.0287 intellectual 0.0287 property 0.0287 list 0.0287 legislate 0.0179 patent))</text>
</query>
<query>
<number>2</number>
<text>#weight(0.5 #weight(1.0 acquisitions))</text>
</query>
<query>
<number>3</number>
<text>#weight(0.5 #weight(0.5 joint 0.5 ventures) 0.5 #weight (0.0168 unite 0.122 joint 0.0053 staff 0.0138 congress 0.0142 committee 0.0049 company 0.0164 state 0.0088 venture 0.0076 force 0.0051 chief))</text>
</query>
<query>
<number>4</number>
<text>#weight(0.5 #weight(0.5 debt 0.5 rescheduling) 0.5 #weight (0.0231 unite 0.0098 sovereign 0.02 govern 0.0148 collateral 0.1613 debt 0.0153 obligate 0.0092 external 0.0231 state 0.0281 crisis 0.0189 public))</text>
</query>
<query>
<number>5</number>
<text>#weight(0.5 #weight(0.5 dumping 0.5 charges))</text>
</query>
```

**Figure 7: Le nouveau fichier de paramètres de la commande IndriRunQuery après l'ajout des termes d'expansion et leurs poids correspondants**



## Résultats expérimentaux de la recherche et discussions :

Comme dans la partie 2, nous avons construit un ensemble de requêtes qui combine les termes de la requête initiale avec les nouveaux termes. Puis nous avons exécuté les deux commandes IndriRunQuery et trec\_eval pour obtenir des résultats que nous rapportons dans les tables qui suivent.

**Tableau 9:** Valeur de MAP pour les différents ensemble de requêtes avec les anchor-log (Modèle de langue avec stemming Porter et stoplist)

MAP	Requête sans expansion	Requête avec expansion
Topic 1-50	0.1554	0.1340
Topic 51-100	0.2262	0.2013
Topic 101-150	0.1630	0.1717

### Observation :

Nous observons du tableau précédent que la valeur du MAP avec expansion en utilisant l'anchor log pour le premier et deuxième ensemble de topic (1-50 et 51-100) a diminué (0.1554 et 0.2262 contre 0.1340 et 0.2013 respectivement). Par contre pour le troisième ensemble de topic (101-150), la valeur du MAP a augmenté de celle sans expansion de requête (0.1717 avec expansion contre 0.1630 sans expansion).

**Tableau 10:** Valeur de P@10 pour les différents ensemble de requêtes avec les anchor-log (Modèle de langue avec stemming Porter et stoplist)

P@10	Requête sans expansion	Requête avec expansion
Topic 1-50	0.1760	0.1820
Topic 51-100	0.4220	0.3700
Topic 101-150	0.3300	0.3480

### Observation :

Pour la variable P@10 l'expansion des requêtes en utilisant l'anchor log a augmenté pour les Topics 1-50 et 101-150 0.1760, 0.3300, contre 0.1820, 0.3480 respectivement. Par contre, on remarque une grande diminution pour le 2em Topics 51-100 0.3700 contre 0.4220.

**Tableau 11:** Valeur du Rappel pour les différents ensemble de requêtes avec les anchor-log (Modèle de langue avec stemming Porter et stoplist)

Rappel	Requête sans expansion	Requête avec expansion
Topic 1-50	0.3897	0.4051
Topic 51-100	0.4273	0.3941
Topic 101-150	0.4343	0.4285

### Observation :

Pour l'ensemble des Topics 51-100, 101-150, la valeur du rappel a diminué considérablement en utilisant l'expansion de requêtes. On a des valeurs de : 0.3941 contre 0.4273 pour les requêtes 51-100, 0.4285 contre 0.4343 pour les requêtes 101-150. L'augmentation n'est observée que dans le premier Topic 1-50 on a 0.3897 contre 0.4051.

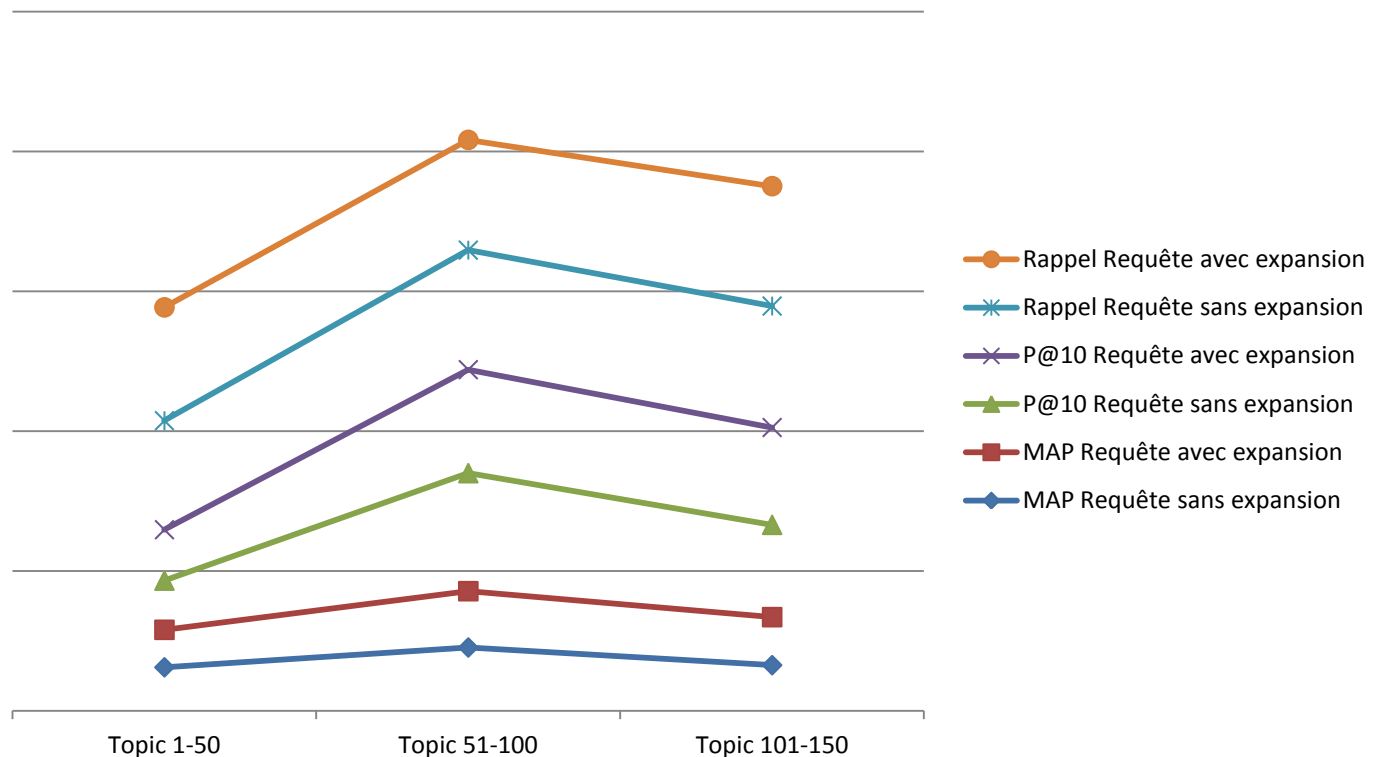


Figure 8: Résumé des variables testées avec et sans expansion pour la méthode de Anchor-Log

### Conclusion partie 3:

L'expansion de la requête n'a pas amélioré considérablement les valeurs de nos variables testées. Pour la précision moyenne, une petite augmentation n'est aperçue que pour le 3eme Topic 101-150 requête à effacer.

La précision pour les 10 premiers documents n'a vu de hausse que pour le Topic 1-50 et 101-150.

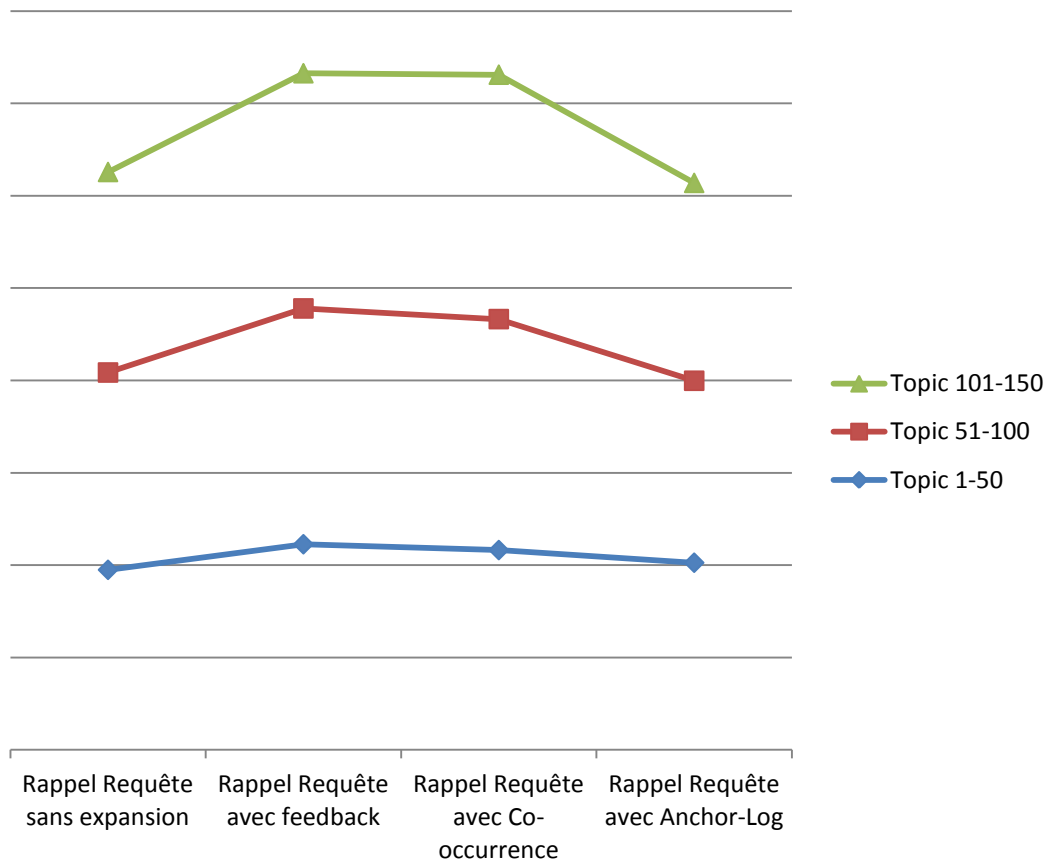
Pour le rappel une fine augmentation pour le 1er Topic 1-50.

Nous pouvons conclure que cette détérioration est due au fait que les clicks anchor ne sont pas liées aux termes de nos requêtes car cette source est une source extérieure. Par conséquent, le contexte de la requête initiale peut être très différent du contexte de l'Anchor log. Prenons l'exemple de la requête 7 du topic 1 qui est « U.S budget deficit », nous trouvons parmi les termes d'expansion le terme « kingdom » qui n'a aucune relation sémantique avec les termes de la requête initiale ou encore pour la requête 23 du même topic qui est « legal repercussions of agrochemical use » avec un terme d'expansion qui est « brother » en plus avec un poids considérable qui est de 0.0484 contre 0.2 le poids d'un terme de la requête initiale. Ce qui peut être triviale car on utilise une source externe pour étendre nos requête

## Comparaison des trois méthodes d'expansion de requêtes

Nous avons voulu faire la comparaison des trois méthodes utilisées. Nous avons choisi à vérifier la valeur de la variable qui nous renvoi le rappel à savoir notre ratio :  $\text{num\_rel\_ret} / \text{num\_rel}$

Rappel	Requête sans expansion	Requête avec expansion		
		Feed-back	Co-occurrence	Anchor-log
Topic 1-50	0.3897	0.4451	0.4325	0.4051
Topic 51-100	0.4273	0.5107	0.5002	0.3941
Topic 101-150	0.4343	0.5094	0.5294	0.4285



**Figure 9: Résumé des variables testées avec et sans expansion pour la méthode de Anchor-Log**

Nous remarquons la variabilité du rappel pour les 3 méthodes comparativement à la requête initiale et cela pour les 3 Topics. D'après le graphique de la *figure 6* le rappel est en croissance depuis la requête initiale pour arriver au maximum pour la méthode de feedback, il décroît considérablement pour la méthode Anchor-Log. Nous pouvons conclure que la méthode feedback a tranché pour une amélioration de rappel et de précision. Ceci peut s'expliquer par le fait que la technique de documents de feedback utilise documents pertinents récupérés lors de la première itération de recherche donc les termes ajoutés vont être des termes extraits de documents pertinents ce qui augmente la chance

que ces termes aient une dépendance sémantique forte avec les termes de la requête initiale. Par opposition à la technique d'Anchor log qui cherche les termes d'expansion dans une ressource extérieure. D'un autre côté, la technique qui utilise la relation de co-occurrence entre termes l'emporte sur la technique d'Anchor log puisqu'elle enrichit la requête initiale avec des termes du vocabulaire de la collection.

## Références

- Expansion sémantique des requêtes :  
[https://www.uclouvain.be/cps/ucl/doc/cental/documents/presentation\\_31\\_03\\_2010.pdf](https://www.uclouvain.be/cps/ucl/doc/cental/documents/presentation_31_03_2010.pdf)
- Wiki d'Indri – dumpindex :  
<http://sourceforge.net/p/lemur/wiki/dumpdoc,%20dumpterm,%20and%20dumpindex/>