MVSim : une architecture pour l'apprentissage multivue de similarités

Gilles Bisson et Clément Grimal

Laboratoire d'Informatique de Grenoble (LIG) – Equipe AMA
{prenom.nom}@imag.fr

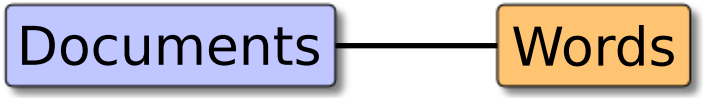
Conférence Francophone sur l'Apprentissage Automatique
Nancy, 25 Mai 2012

UNIVERSITÉ DE GRENOBLE

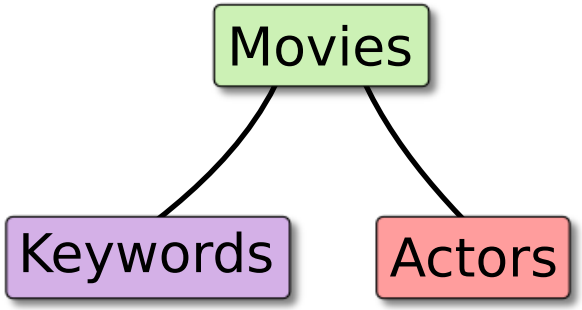
Plan

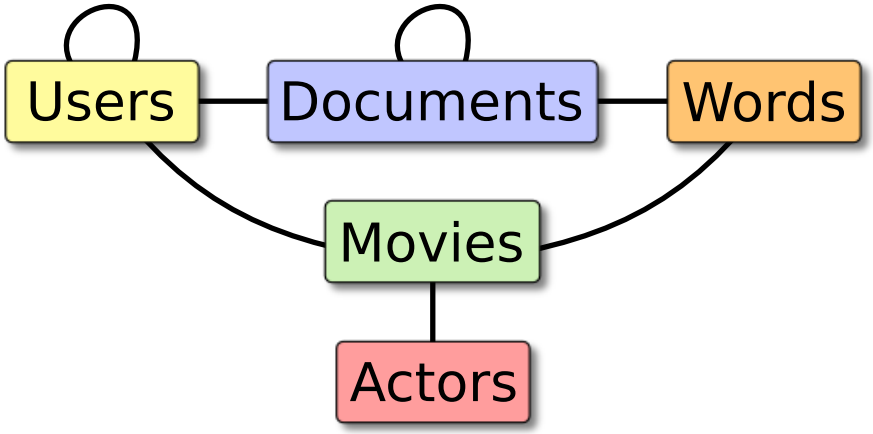
Données multivues

Pas seulement 2 types d'objets (Documents/ Mots) liés par 1 relation...



...mais de multiple types d'objets, liés par de multiple relations.





Notations

Entrée

- N types d'objets, notés T_i, avec n_i instances
- M matrices de relation (vues), notées $\mathbf{R}_{ij}$, décrivant la relation entre les instances de T_i et les instances de T_j, de taille n_i by n_j

Sortie

- N matrices de similarités, notées $\mathbf{S}_i$, dont les éléments sont les mesures de similarités entre toutes les paires des instances de T_i

L'algorithme χ-Sim

Modèle

$\mathbf{R}$: une matrice documents/mots, de n_1 lignes et n_2 colonnes
Nous voulons calculer :

- $\mathbf{S_1}$: matrice de similarité carrée et symétrique des documents, de taille $n_1 \times n_1$, à valeur dans $[0, 1]$
- $\mathbf{S_2}$: matrice de similarité carrée et symétrique des mots, de taille $n_2 \times n_2$, à valeur dans $[0, 1]$

Idée de base

- Deux documents sont similaires s'ils contiennent des mots similaires.
- Deux mots sont similaires s'ils apparaissent dans des documents similaires.

Nous allons construire conjointement les matrices $\mathbf{S_1}$ et $\mathbf{S_2}$.

χ-Sim – Equations

Calcul de $\mathbf{S}_1$ basé sur $\mathbf{S}_2$:

$$\mathbf{S}_1 = \mathbf{R}^{\circ k} \times \mathbf{S}_2 \times \left(\mathbf{R}^{\circ k}\right)^{\mathrm{T}}$$

$$\forall a, b \; (\mathbf{S}_1)_{ab} \leftarrow \left(\frac{(\mathbf{S}_1)_{ab}}{\sqrt{(\mathbf{S}_1)_{aa} \times (\mathbf{S}_1)_{bb}}} \right)^{1/k}$$

avec $\mathbf{R}^{\circ k}$ désignant l'opération de mise à la puissance k des éléments de $\mathbf{R}$

χ-Sim – Equations

Calcul de $\mathbf{S}_1$ basé sur $\mathbf{S}_2$:

$$\mathbf{S}_1 = \mathbf{R}^{\circ k} \times \mathbf{S}_2 \times (\mathbf{R}^{\circ k})^{\mathrm{T}}$$

$$\forall a, b \; (\mathbf{S}_1)_{ab} \leftarrow \left(\frac{(\mathbf{S}_1)_{ab}}{\sqrt{(\mathbf{S}_1)_{aa} \times (\mathbf{S}_1)_{bb}}} \right)^{1/k}$$

Et calcul de $\mathbf{S}_2$ basé sur $\mathbf{S}_1$:

$$\mathbf{S}_2 = (\mathbf{R}^{\circ k})^{\mathrm{T}} \times \mathbf{S}_1 \times \mathbf{R}^{\circ k}$$

$$\forall a, b \; (\mathbf{S}_2)_{ab} \leftarrow \left(\frac{(\mathbf{S}_2)_{ab}}{\sqrt{(\mathbf{S}_2)_{aa} \times (\mathbf{S}_2)_{bb}}} \right)^{1/k}$$

avec $\mathbf{R}^{\circ k}$ désignant l'opération de mise à la puissance k des éléments de $\mathbf{R}$

χ-Sim – Algorithme

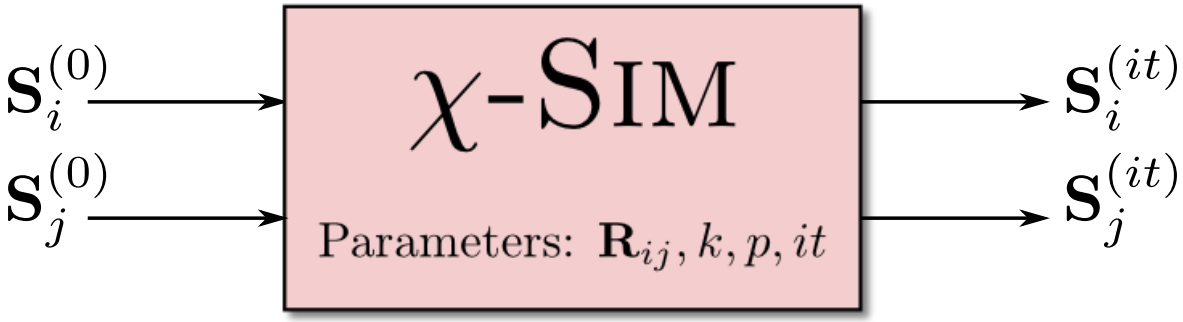
Entrées : la matrice $\mathbf{R}$, et les paramètres k, p, it
Sorties : les matrices de similarités $\mathbf{S}_1$ et $\mathbf{S}_1$

1. Initialisation de $\mathbf{S_1}^{(0)}$ et $\mathbf{S_2}^{(0)}$ par la matrice identité
2. Pour $t \in [1..it]$, mise à jour des 2 matrices de similarités :
   3. Mise à jour de $\mathbf{S_1}^{(t)}$ en utilisant $\mathbf{S_2}^{(t-1)}$
   4. Elagage des $p\%$ valeurs les plus faibles de $\mathbf{S_1}$
   5. Mise à jour de $\mathbf{S_2}^{(t)}$ en utilisant $\mathbf{S_1}^{(t-1)}$
   6. Elagage des $p\%$ valeurs les plus faibles de $\mathbf{S_2}$

En pratique, $it = 4$ est satisfaisant.

χ-Sim – Vue fonctionnelle

On considère une matrice de relation $\mathbf{R}_{ij}$ décrivant la relation entre les instances de T_i et celles de T_j.

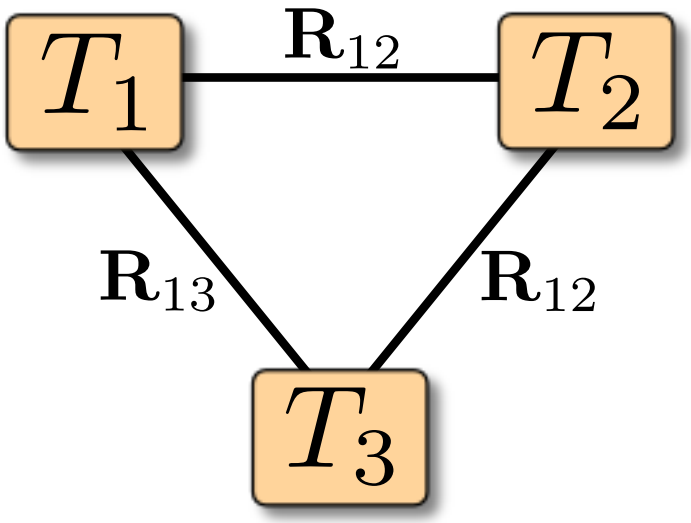


Il est possible d'initialiser χ-Sim avec des connaissances *a priori* sur les similarités *via* $\mathbf{S}_i^{(0)}$ et $\mathbf{S}_j^{(0)}$.

Finalement, on obtient les matrices de similarités $\mathbf{S}_i^{(it)}$ et $\mathbf{S}_j^{it)}$ à partir de cette matrice $\mathbf{R}_{ij}$ traité par χ-Sim avec les paramètres it, k, p.

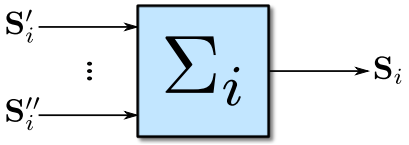
L'architecture MVSim

N types d'objets : $\{T_i\}$
M relations : $\{\mathbf{R}_{ij}\}$

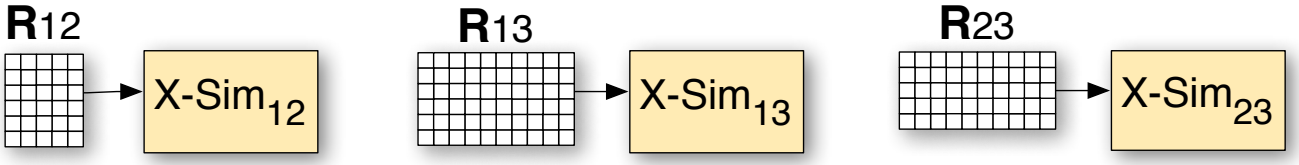


MVSim – L'architecture

- à chaque T_i, on associe une fonction d'aggrégation Agg_i

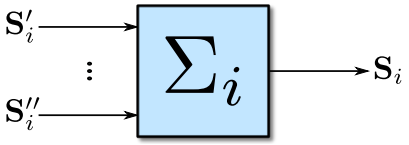


- à chaque $\mathbf{R}_{ij}$, on associe une instance de l'algorithme χ-Sim : $\chi\text{-Sim}_{ij}$

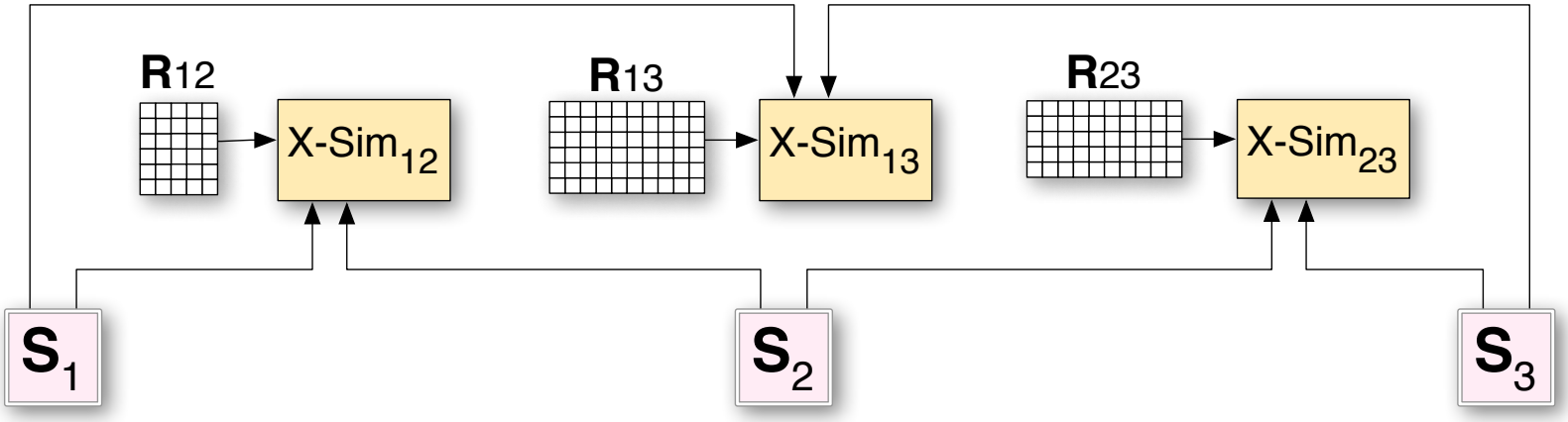


MVSim – L'architecture

- à chaque T_i, on associe une fonction d'aggrégation Agg_i

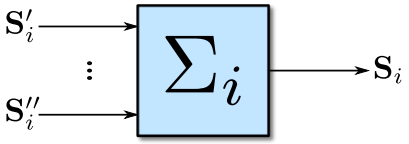


- à chaque $\mathbf{R}_{ij}$, on associe une instance de l'algorithme χ-Sim : $\chi\text{-Sim}_{ij}$

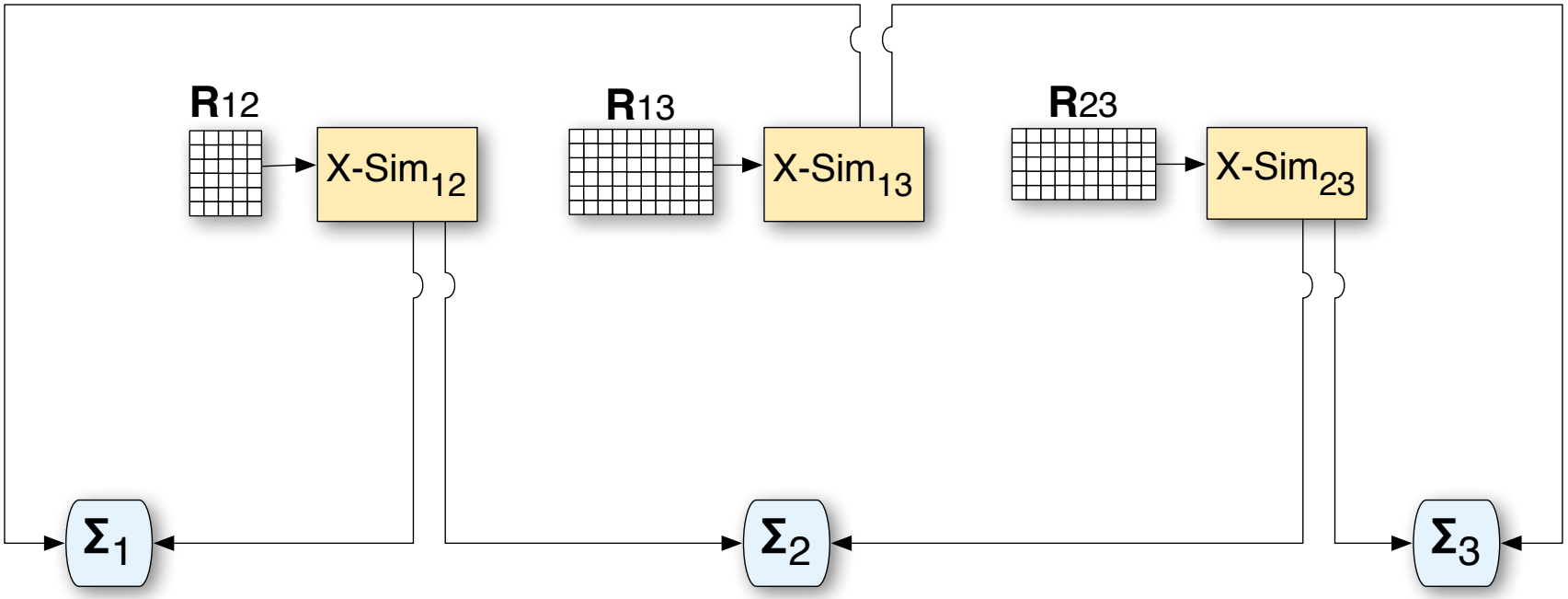


MVSim – L'architecture

- à chaque T_i, on associe une fonction d'aggrégation Agg_i

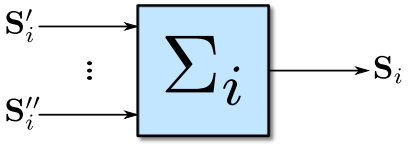


- à chaque $\mathbf{R}_{ij}$, on associe une instance de l'algorithme χ-Sim : $\chi\text{-Sim}_{ij}$

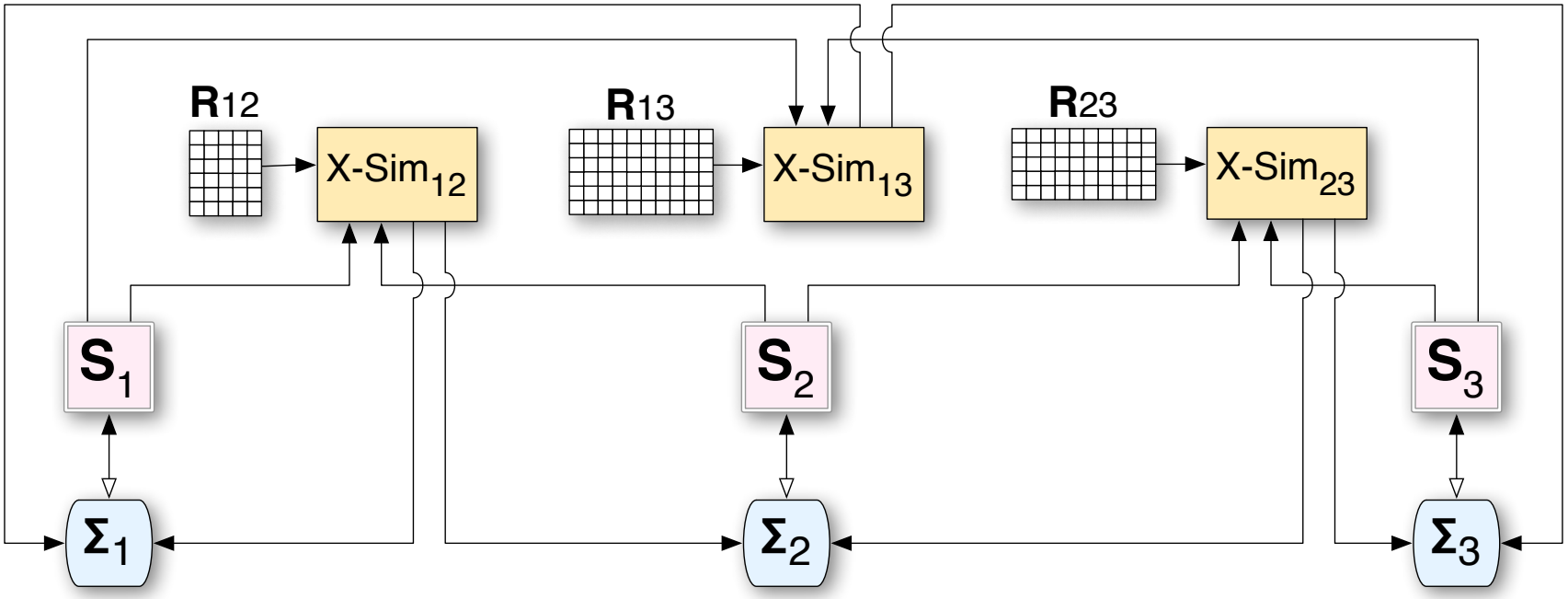


MVSim – L'architecture

- à chaque T_i, on associe une fonction d'aggrégation Agg_i

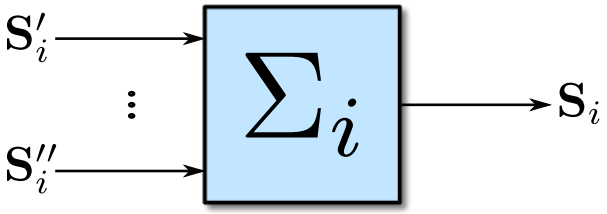


- à chaque $\mathbf{R}_{ij}$, on associe une instance de l'algorithme χ-Sim : $\chi\text{-Sim}_{ij}$



MVSim – Fonction d'agrégation

Combiner plusieurs matrices de similarité (concernant les mêmes objets) en une seule



$$\text{Agg}_i = \frac{\mathbf{S}_i^{(t-1)} + \lambda^t \times F(\mathbf{S}'_i, \mathbf{S}''_i, \dots)}{1 + \lambda^t}$$

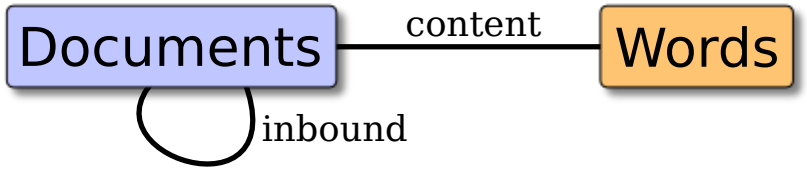
λ : paramètre d'amortissement
F : fonction de fusion (minimum, maximum, moyenne...)

La fonction F étant bornée et λ^t étant exponentiellement décroissant avec t : l'architecture converge

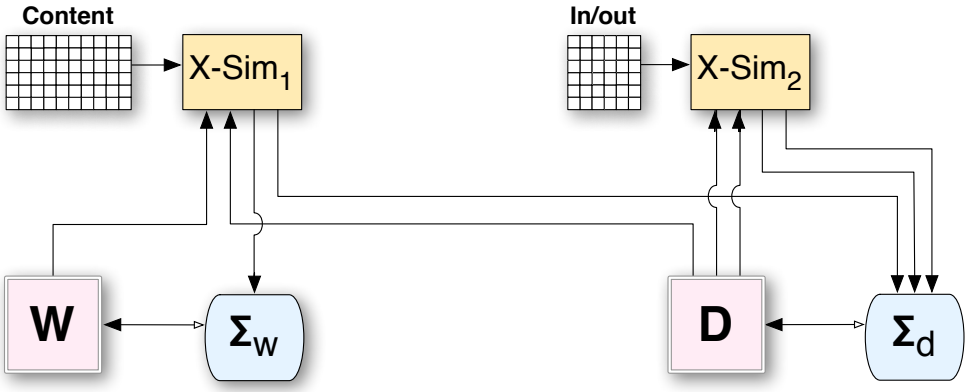
MVSim – Exemple avec des données web

2 types d'objets : documents et mots, mais avec 2 relations

- Content : matrice documents / mots classique
- Inbound : matrice de citations entre documents



Nous pouvons également considérer les liens sortants (gratuit avec χ-Sim !) :



MVSim – Dynamique de l'architecture

Asynchrone

- Ordre d'exécution pour les χ-Sim$_{ij}$ et les Agg$_i$
- Principal problème : favorise le χ-Sim$_{ij}$ exécuté en dernier

Synchrone

On alterne entre 2 étapes :

- exécution de tous les χ-Sim$_{ij}$ (en parallèle si possible)
- exécution de toutes les Agg$_i$

jusqu'à convergence (ou un nombre maximum d'itérations)

Dans nos expérimentations, nous utilisons la version synchrone.

MVSim – Algorithme

Les nombre d'itérations des instances de χ-Sim est fixé à 1 et on utilise un nombre total d'itérations : I_G

Entrées : les matrices $\mathbf{R}_{ij}$, et les paramètres I_G, λ, k, p
Sorties : les matrices de similarités $\mathbf{S}_i$

1. Initialisation des $\mathbf{S}_i^{(0)}$ par la matrice identité
2. Pour $t \in [1..I_G]$:
    3. Exécuter χ-Sim$_{ij}$ avec $it = 1, k, p$
    4. Mise à jour des $\mathbf{S}_i^{(t)}$ par les Agg$_i$

MVSim – Complexité et parallélisation (1)

On considère une matrice de relation $\mathbf{R}$ de taille $n \times p$, avec $p > n$

Complexité de χ-Sim

Produit de 3 matrices : $\mathcal{O}(np^2 + n^2p) = \mathcal{O}(np^2)$

Complexité de Agg_i

Pour des opérations terme à terme : $\mathcal{O}(p^2)$

Complexité de MVSim

Si nous avons M matrices de relations, on peut utiliser M cœurs (ou CPU) pour les calculs...
Donc la complexité temporelle reste en $\mathcal{O}(np^2)$.

MVSim – Complexité et parallélisation (2)

Exemple particulier

- Une matrice documents / mots de taille $n \times p$
- on veut calculer les similarités uniquement entre documents
- $p >> n$

On peut diviser les mots en M sous-ensembles, obtenant ainsi M matrices. Ensuite, on utilise MVSim (sur M cœurs), avec un gain :

- en complexité temporelle en $1/M^2$
- en complexité spatiale en $1/M$

Experimentations

Evaluer les performances de l'architecture MVSim :

- sur des jeux de données multivues, en se comparant à :
  - des méthodes classiques (monovue)
  - d'autres approches multivues
- sur des jeux de données monovues, en utilisant l'approche par découpage

Méthodologie

On dispose des vraies classes des instances pour un type d'objets (documents ou films). On mesure l'adéquation entre les vraies classes, et le résultat de la classification.

Méthodes

Méthodes monovue

- Cosinus
- LSA [Deerwester et al.(1990)]
- SNOS [Liu et al.(2004)]
- CTK [Yen et al.(2009)]
- ITCC [Dhillon et al.(2003)]
- χ-Sim [Hussain et al.(2010)]

Pour les mesures de similarités, on utilise une Classification Ascendante Hiérarchique (CAH), avec le critère de Ward pour obtenir la classification.

Méthodes multivues

- MVKM : extension multivue des k-moyennes [Drost et al.(2006)]
- MVSC : classification spectrale multivue [Kumar & Daume III (2011)]

Tests multivues – Description des jeux de données

- 1 jeu extrait d'IMDb avec 2 vues sur les films : description par les acteurs, et descriptions par des mots-clés
- 6 jeux de documents avec 2 vues : 1 vue sur le contenu (documents / mots) et 1 vue sur les citations (entre documents)
- 1 jeu multilingue de documents avec 5 vues, correspondant à 5 langages

	Films	Mots-clés	Acteurs	Classes
IMDb	617	1878	1398	17

	Doc.	Mots	Liens	Classes
Cora	2708	1433	5429	7
Citeseer	3312	3703	4732	6
Cornell	195	1703	569	5
Texas	187	1703	578	5
Washington	230	1703	783	5
Winconsin	265	1703	938	5

	Doc.	Mots	Classes
Reuters	1200	2500 dans chaque vue	6

Tests multivues – Résultats

Jeu (mesure)	meilleur monovue	MVKM	MVSC	MVSim
Movie (Pr)	0,290	-	-	**0,347**
Cora (Pr)	0,634	-	-	**0,697**
Citeseer (Pr)	0,608	-	-	**0,635**
Cornell (Pr)	0,631	-	-	**0,692**
Texas (Pr)	**0,722**	-	-	0,615
Washington (Pr)	0,652	-	-	**0,657**
Wisconsin (Pr)	**0,675**	-	-	**0,675**
Citeseer (En)	1,27	1,60	-	**1,07**
Reuters (NMI)	0,320	-	0,388	**0,422**

Pour l'entropie (En), une plus petite valeur indique un meilleur résultat.

- Sur 5 (sur 8) jeux de données, le meilleur résultat monovue est obtenu par χ-Sim
- A part pour Texas, MVSim obtient de meilleurs résultats que les méthodes monovues
- MVSim obtient également de meilleurs résultats que les approches multivues

Tests de découpage – Description des jeux de données

Sous jeux de données créés à partir de la collection de Newsgroup NG20.

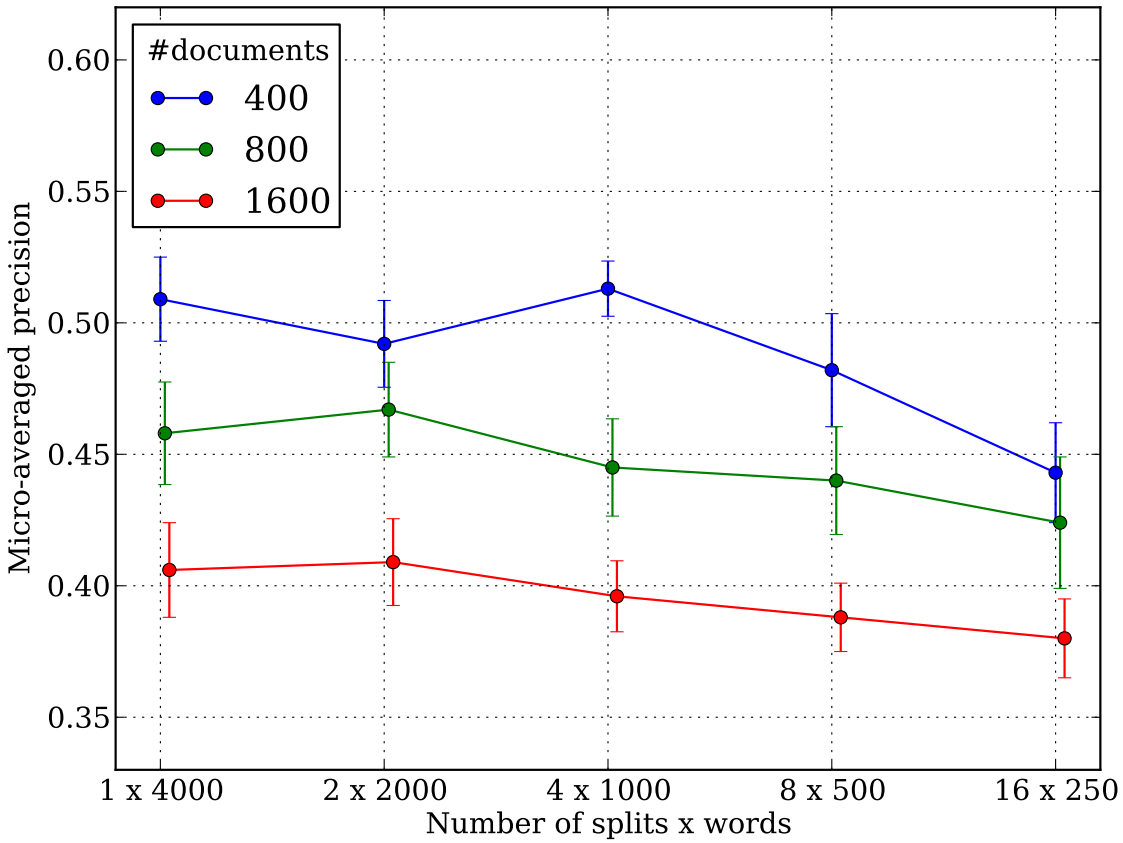
- 10 classes : comp.graphics, misc.forsale, rec.autos, rec.sport.baseball, rec.sport.hockey, sci.crypt, sci.med, sci.space, soc.religion.christian, talk.politics.mideast
- Nombre de documents : 400, 800 et 1600
- Nombre de mots : de 500 à 4000 (sélectionnés avec PAM)

On veut répondre à 2 questions :

1. Pour un nombre de mots constant, comment évolue la performance en augmentant le nombre de divisions ?
2. Peut-on améliorer la classification en considérant plus de mots à temps constant ?

Tests de découpage (2)

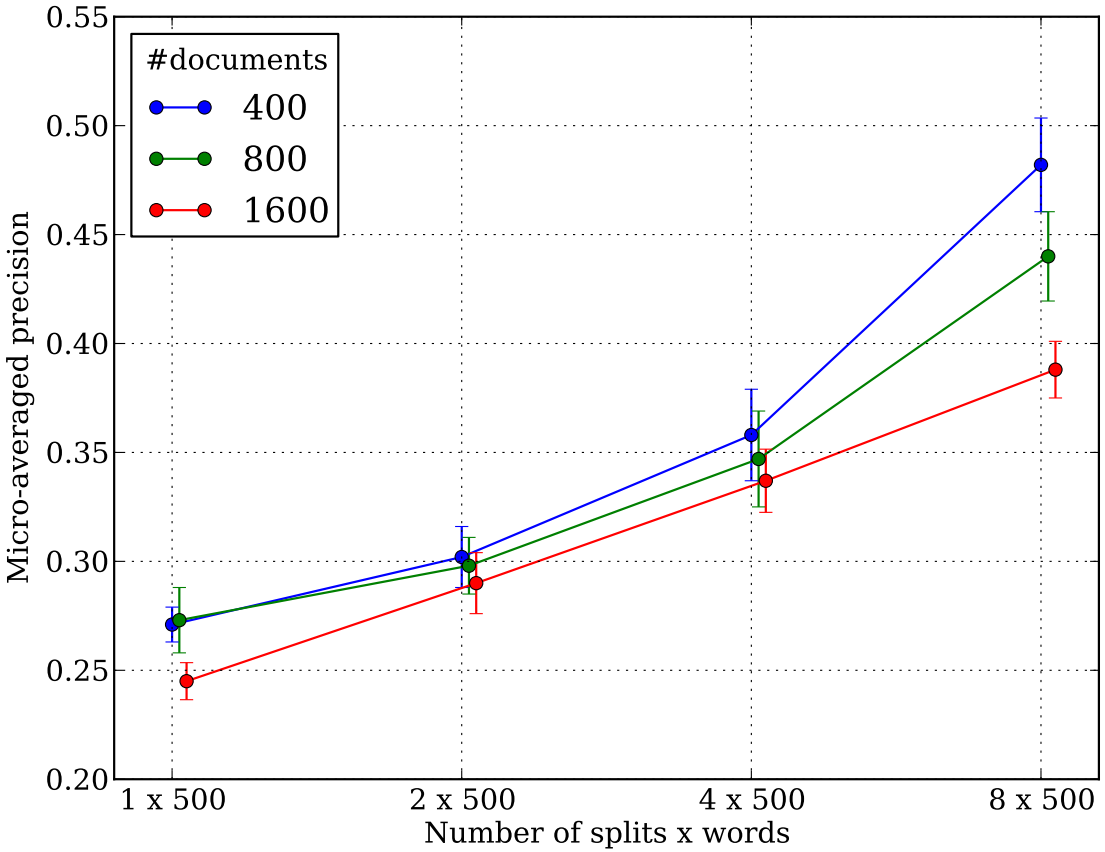
A nombre total de mots constant...



...on note une légère dégradation de la performance, mais le temps de calcul a été drastiquement réduit, en $1/M^2$.

Tests de découpage (3)

A temps de calcul constant (nombre de mots par sous-matrices constant)...



...on remarque qu'on peut nettement améliorer les résultats.

Conclusion

Conclusion

Basée sur χ-Sim, nous avons proposé l'architecture MVSim :

- propriétés de convergence et de passage à l'échelle intéressantes
- meilleurs résultats que les méthodes monovues et multivues
- résultats prometteurs pour traiter des problèmes de grandes dimensions

Perspectives

- Améliorer la formalisation de l'approche
- Compléter la validation expérimentale, avec plus de comparaisons avec d'autres méthodes multivues

Une piste intéressante :

- Découpage des lignes et des colonnes d'une grande matrice,

Merci de votre attention.

Des questions ? Suggestions ? Remarques ?

References

Kumar, A. and Daume III, H. A co-training approach for multi-view spectral clustering. *ICML*, pp. 393–400, 2011.

Drost, I., Bickel, S., and Scheffer, T. Discovering communities in linked data by multi-view clustering. *From Data and Information Analysis to Knowledge Engineering*, pp. 342–349, 2006.

S. Deerwester, S. T. Dumais, G. W. Furnas, Thomas, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41 :391–407, 1990.

I. S. Dhillon, S. Mallela, and D. S. Modha. Information-theoretic co-clustering. In *Proceedings of the 9th ACM SIGKDD*, pages 89–98, 2003.

S. F. Hussain, C. Grimal, and G. Bisson. An improved co-similarity measure for document clustering. In *Proceedings of the 9th ICMLA*, 2010.

N. Liu, B. Zhang, J. Yan, Q. Yang, S. Yan, Z. Chen, F. Bai, and W. ying Ma. Learning similarity measures in non-orthogonal space. In *Proceedings of the 13th ACM CIKM*, pages 334–341. ACM Press, 2004.

L. Yen, F. Fouss, C. Decaestecker, P. Francq, and M. Saerens. Graph nodes clustering with the sigmoid commute-time kernel : A comparative study. *Data Knowl. Eng.*, 68(3) :338–361, 2009.

C. C. Aggarwal and A. Hinneburg, and D. A. Keim. On the Surprising Behavior of Distance Metrics in High Dimensional Space. *Lecture Notes in Computer Science*, 420–434, 2001.

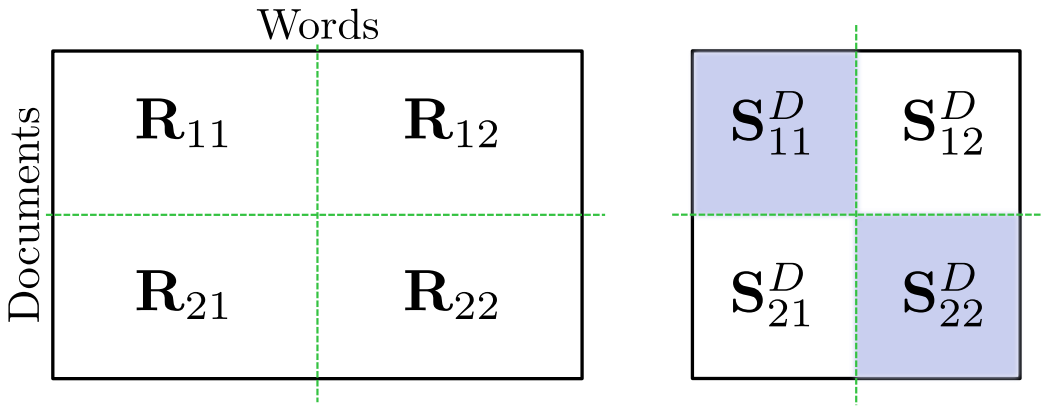
Multiway Splitting

Split both rows and columns.

Main problem : if the documents are split in multiple sets, you don't have the similarities between all pairs of documents...

Proposed idea : reconstruct these similarities based on the words similarities (even if words are split as well)

Example : $\mathbf{R}$ is split in $\mathbf{R}_{11}$, $\mathbf{R}_{12}$, $\mathbf{R}_{21}$ and $\mathbf{R}_{22}$.



$$\mathbf{S}^D_{12} = \mathbf{R}_{11}\,\mathbf{S}^W_{11}\,\mathbf{R}^{\mathrm{T}}_{21} + \mathbf{R}_{12}\,\mathbf{S}^W_{22}\,\mathbf{R}^{\mathrm{T}}_{22}$$

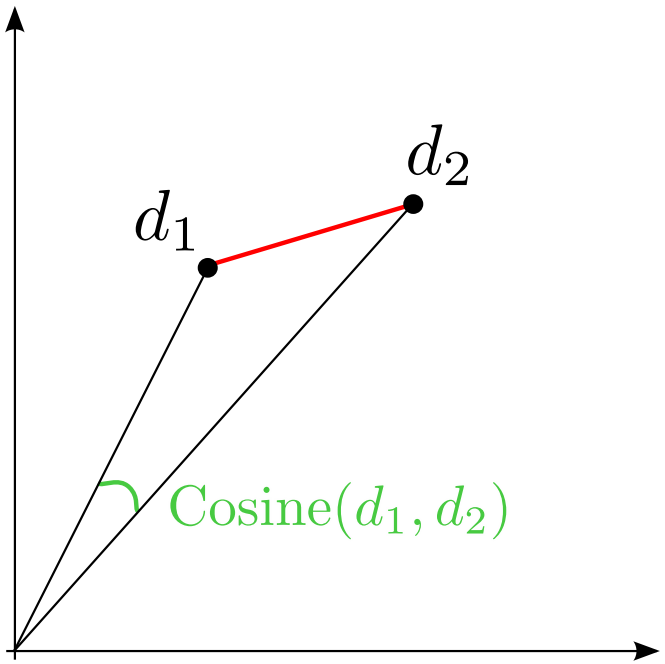
And a normalization...

Similarities / Distances

Euclidean distance, Manhattan distance, Minkowski distance...

For documents :

$$\text{Cosine}(d_1, d_2) = \frac{d_1 \cdot d_2}{\|d_1\| \, \|d_2\|}$$



Similarity between two objects

- Classical approach : similarity = f(shared features)

$$\mathrm{Sim}(\mathbf{r}_{i:}, \mathbf{r}_{j:}) = \mathrm{F_s}(r_{i1}, r_{j1}) + \cdots + \mathrm{F_s}(r_{ic}, r_{jc})$$

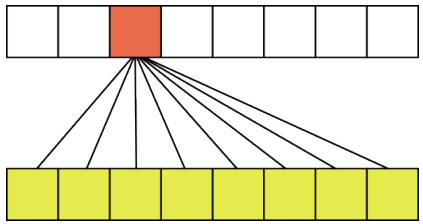
with $\mathrm{F_s}$ a similarity function (absolute difference, product, etc.).

- Using $\mathbf{S_2}$ (usually, $(S_2)_{ll} = 1$) :

$$\mathrm{Sim}(\mathbf{r}_{i:}, \mathbf{r}_{j:}) = \sum_{l=1}^{c} \mathrm{F_s}(r_{il}, r_{jl}) \times (S_2)_{ll}$$

- Now comparing every pairs of features :

$$\mathrm{Sim}(\mathbf{r}_{i:}, \mathbf{r}_{j:}) = \sum_{l=1}^{c} \sum_{n=1}^{c} \mathrm{F_s}\left(r_{il}, r_{jn}\right) \times (S_2)_{ln}$$



Pseudo-norm k

- If $\mathrm{F_s}(r_{ij}, r_{kl}) = r_{ij} \times r_{kl}$:

$$\mathrm{Sim}(\mathbf{r}_{i:}, \mathbf{r}_{j:}) = \mathbf{r}_{i:} \times \mathbf{S_2} \times \mathbf{r}_{j:}^{\mathrm{T}}$$

- We introduce a pseudo-norm k (see [Aggarwal et al.(2001)]) :

$$\mathrm{Sim}^k(\mathbf{r}_{i:}, \mathbf{r}_{j:}) = \sqrt[k]{(\mathbf{r}_{i:})^k \times \mathbf{S_2} \times \left(\mathbf{r}_{j:}^{\mathrm{T}}\right)^k} = \langle \mathbf{r}_{i:}, \mathbf{r}_{j:} \rangle_{\mathbf{S_2}}^k$$

$\rightarrow$ we have $\|\mathbf{r}_{i:}\|_{\mathbf{S_2}}^k = \sqrt{\langle \mathbf{r}_{i:}, \mathbf{r}_{i:} \rangle_{\mathbf{S_2}}^k}$

- Then we need to normalize this similarity :

$$(S_1)_{ij} = \frac{\sqrt[k]{(\mathbf{r}_{i:})^k \times \mathbf{S_2} \times \left(\mathbf{r}_{j:}^{\mathrm{T}}\right)^k}}{\mathcal{N}(\mathbf{r}_{i:}, \mathbf{r}_{j:})} \in [0, 1]$$

Generic form

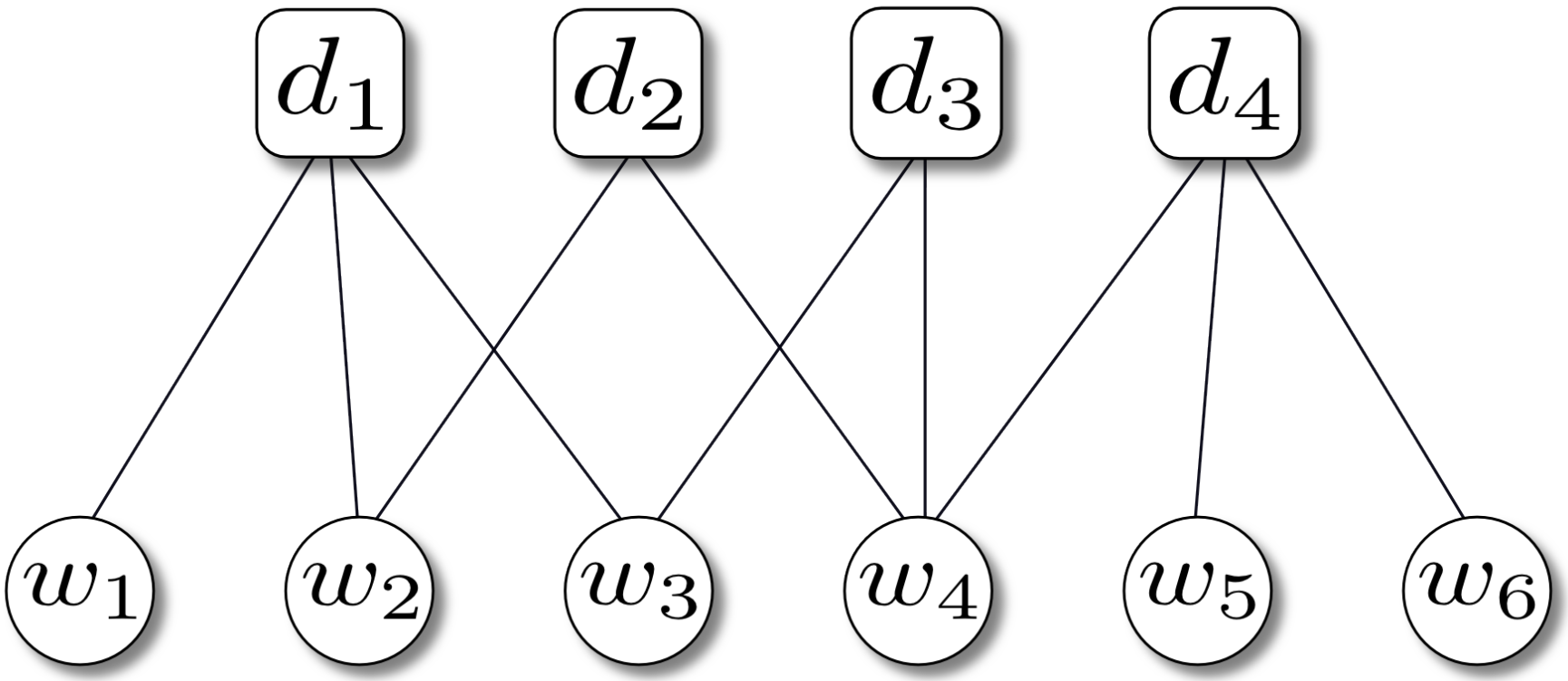
- Now :

$$(S_1)_{ij} = \frac{\sqrt[k]{(\mathbf{r}_{i:})^k \times \mathbf{S_2} \times (\mathbf{r}_{j:}^{\mathrm{T}})^k}}{\mathcal{N}(\mathbf{r}_{i:}, \mathbf{r}_{j:})} = \frac{\langle \mathbf{r}_{i:}, \mathbf{r}_{j:} \rangle_{\mathbf{S_2}}^k}{\mathcal{N}(\mathbf{r}_{i:}, \mathbf{r}_{j:})}$$

- With special values for k, $\mathbf{S_2}$ and $\mathcal{N}$, we have :
  - Jaccard : $\mathbf{S_2} = \mathbf{I}$, $k = 1$, $\mathcal{N} = \|\mathbf{r}_{i:}\|1 + \|\mathbf{r}_{j:}\|1 - \mathbf{r}_{i:}\mathbf{r}_{j:}^{\mathrm{T}}$
  - Dice : $\mathbf{S_2} = 2\mathbf{I}$, $k = 1$, $\mathcal{N} = \|\mathbf{r}_{i:}\|1 + \|\mathbf{r}_{j:}\|1$
  - Generalized Cosine : $\mathbf{S_2} > 0$, $k = 1$, $\mathcal{N} = \|\mathbf{r}_{i:}\|_{\mathbf{S_2}} \times \|\mathbf{r}_{j:}\|_{\mathbf{S_2}}$
  - χ-Sim ^{k} : $\mathcal{N} = \|\mathbf{r}_{i:}\|_{\mathbf{S_2}}^k \times \|\mathbf{r}_{j:}\|_{\mathbf{S_2}}^k$

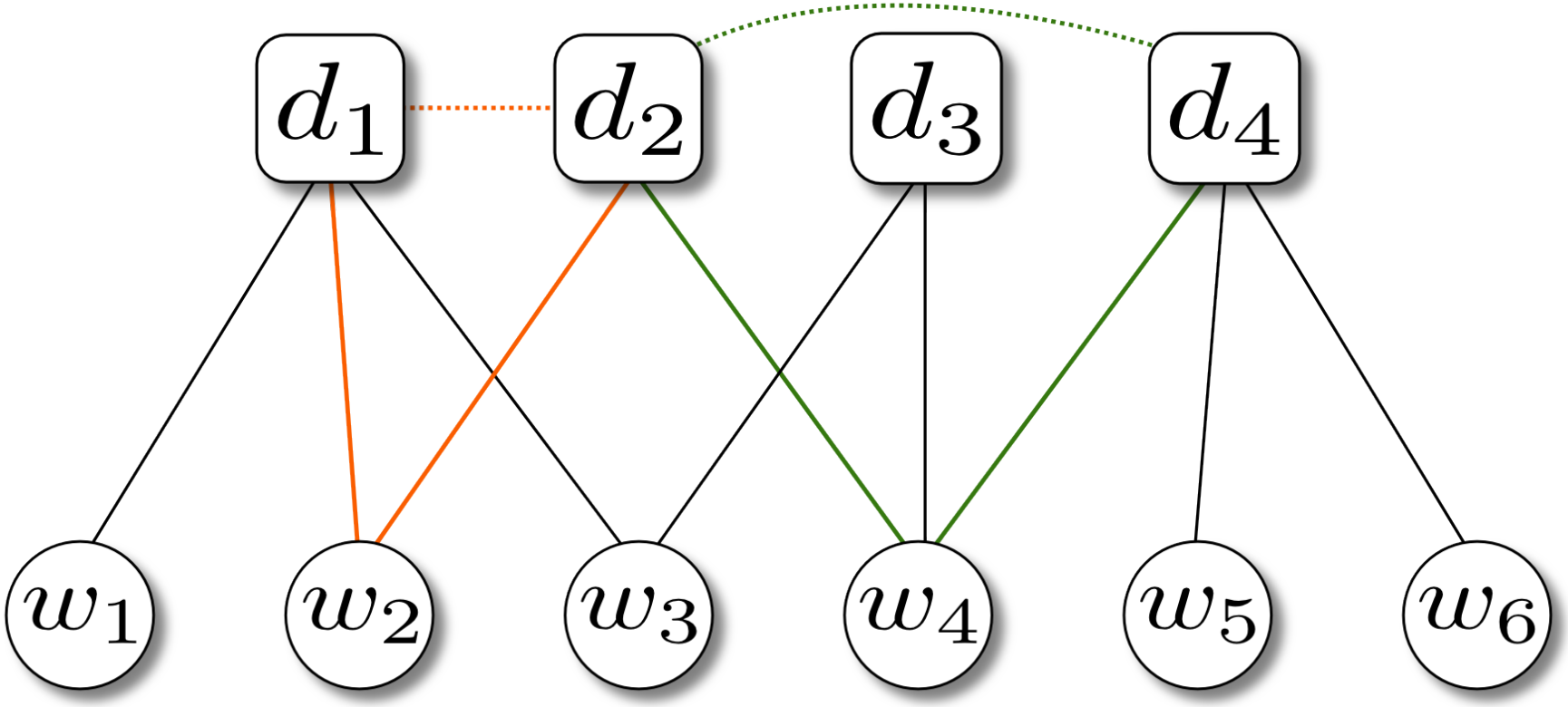
Meaning of an iteration

Bi-partite graph representing a simple corpus



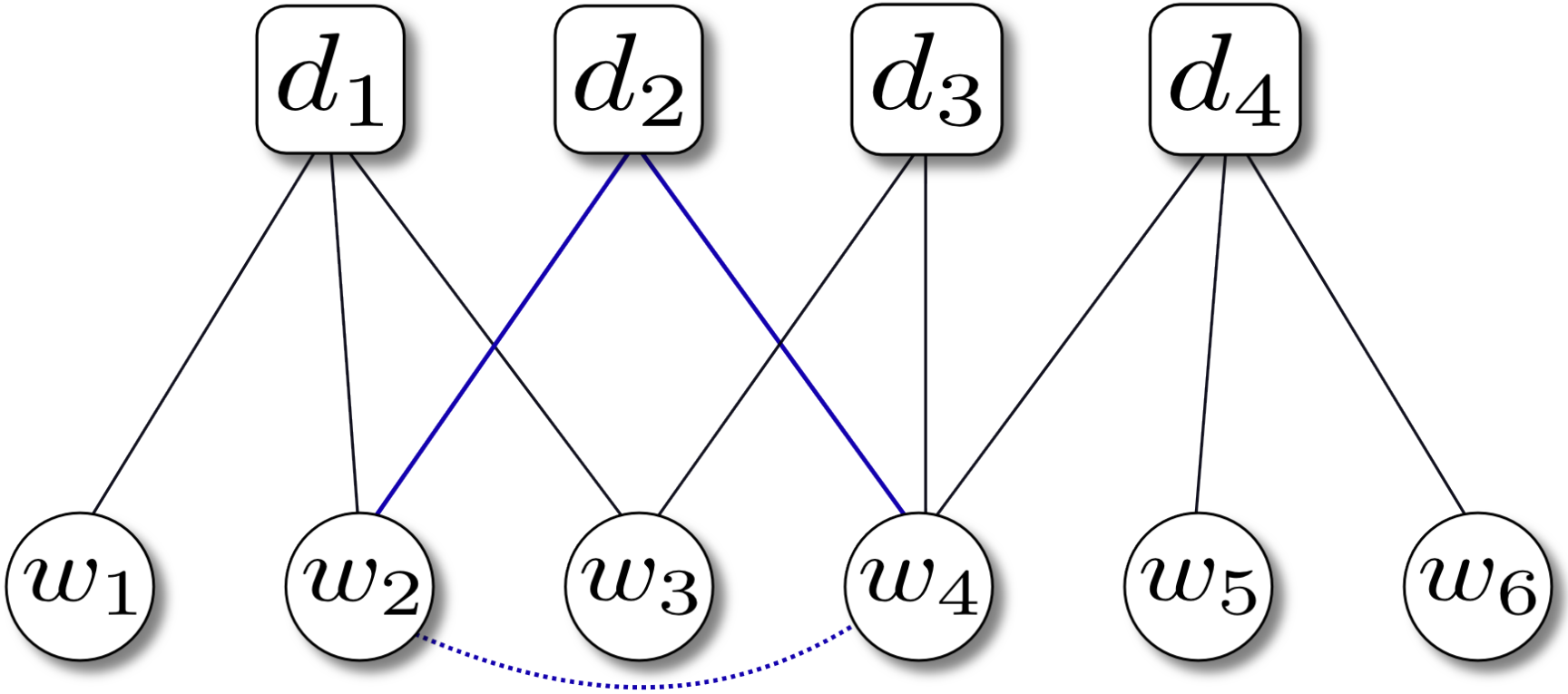
Meaning of an iteration

First iteration : $(S_1)_{12} > 0$ and $(S_1)_{24} > 0$, but $(S_1)_{14} = 0$



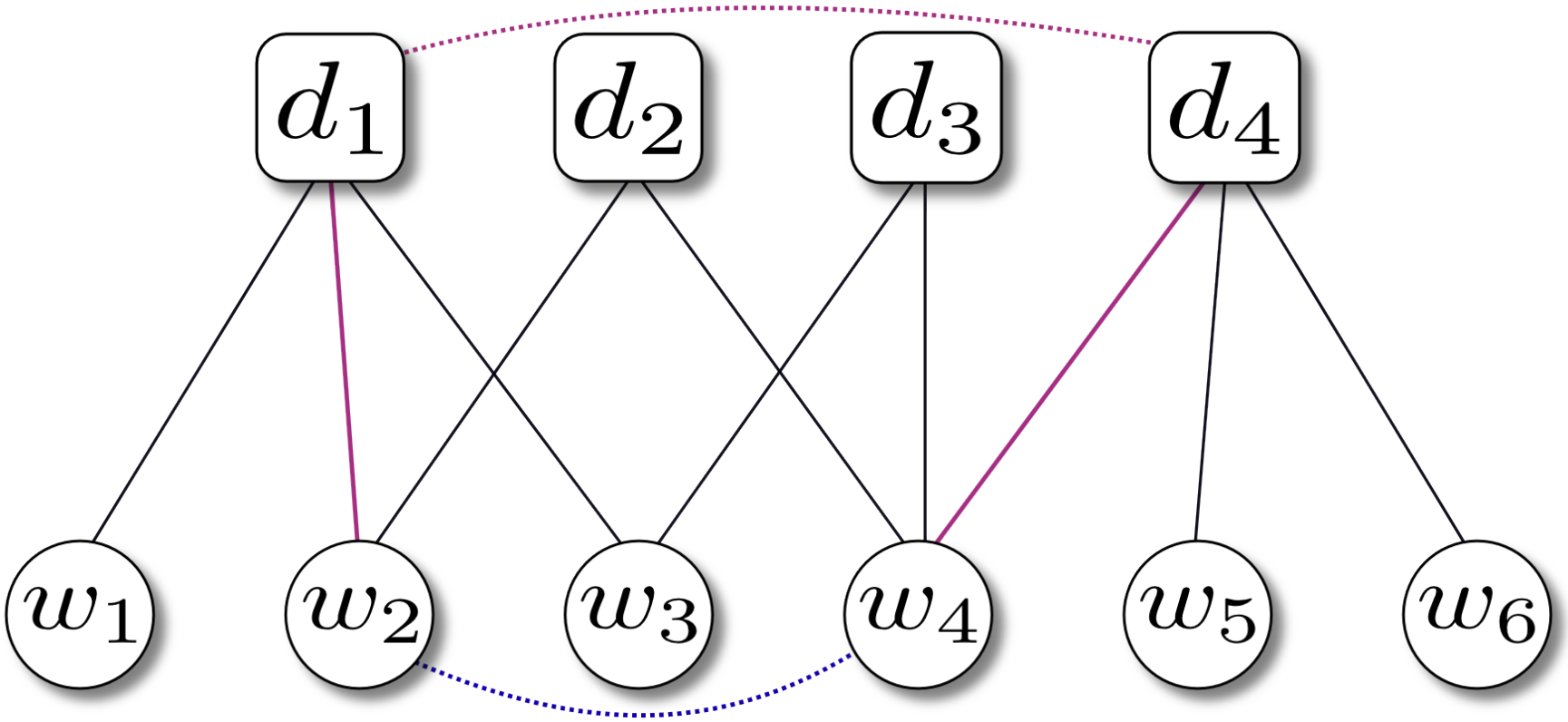
Meaning of an iteration

Second part of the first iteration : $(S_2)_{24} > 0$



Meaning of an iteration

Second iteration : through $(S_2)_{24}$, now $(S_1)_{14} > 0$



Pruning parameter p

In such a corpus...

Many words are not specific enough, and creates a lot of irrelevant similarities. These similarities can be considered as noise.

Example : Astronomy / Mythology

The word *Hercules* can appear once in an astronomy document, and "link" it to all mythology documents dealing with greek heroes...

How to deal with it ?

Hypothetis : these irrelevant similarities are small.
$\rightarrow$ At each iteration, we remove the smallest $p\%$ of the similarity matrices.

Methodology (1)

We have ground-truth labels for the instances of one type of objects (documents or movies).
Measure the match between our clusters and the labels...

Measures

- Pr (micro-averaged Precision) : the percentage of documents correctly clustered (the higher the better)
- NMI (Normalized Mutual Information) : between the actual and the predicted clusterings (the higher the better)
- En (Entropy) : the needed information to encode the actual label knowing the predicted cluster (the lower the better)

Axis (2) – Consensus clustering

Instead of having similarity matrices circulating in the architecture, we can have set of clusters...

We would thus need to replace :

- χ-Sim by a clustering algorithm that can be initialized
- Agg_i by a method to find a consensus clustering

Example : use k-means and aggregate multiple results by taking the barycenter of the seeds (need to be able to find the best matching)

If we want to process to converge, we need to study the Agg_i functions carefully...

Axis (3) – Supervised learning

An extension of χ-Sim has already been used for supervised learning.
Main used strategy :

- Decrease similarities between objects that are not in the same class

For the multi-view setting, we could integrate this strategy in the Agg_i after the merging of the similarity matrices.

Axis (2) + Axis (3) – Ensemble learning

We could use MVSim to do ensemble learning by :

- replacing the χ-Sim by a supervised learning method
- replacing the Agg_i by ensemble a learning approach

Can MVSim be generalized to describe existing works ?

Splitting tests – Results

	M2	M5	M10	NG1	NG2	NG3
MV (1×500)	**0.745**	0.738	**0.507**	**0.782**	0.642	0.509
MV (2×500)	0.710	0.746	0.474	0.629	0.668	**0.609**
MV (4×500)	0.653	**0.756**	0.463	0.546	**0.681**	0.595
MV (1×1000)	**0.745**	0.738	**0.507**	**0.809**	0.680	0.582
MV (2×1000)	0.719	0.769	0.499	0.649	**0.718**	**0.633**
MV (4×1000)	0.643	**0.784**	0.491	0.573	0.689	0.631
χ-Sim (2000)	0.81	0.79	0.55	0.81	0.72	0.64

- For a given numbers of words, the precision decrease with the number of split (2×500 versus 1×1000)
- For a given numbers of words per instance of χ-Sim, we can improve the result by using more words on the biggest datasets (NG2 and NG3)

Trade-off between number of matrices and number of words.