Calcul décentralisé de dictionnaires visuels pour l'indexation multimédia dans les bases de données réparties sur les réseaux

Jérôme Fellus, David Picard, Philippe-Henri Gosselin

Calcul décentralisé de dictionnaires visuels pour l'indexation multimédia dans les bases de données réparties sur les réseaux

Jérôme Fellus[1] David Picard[1] Philippe-Henri Gosselin[2]

[1] ETIS, UMR 8051, ENSEA - Université de Cergy-Pontoise - CNRS
[2] INRIA, Rennes Bretagne Atlantique

6 av du Ponceau, F-95000 Cergy-Pontoise
{prenom}.{nom}@ensea.fr

Résumé

Dans le souci de permettre le passage à l'échelle des méthodes d'indexation par le contenu devant l'explosion de la quantité de contenu disponible, nous présentons un nouveau système décentralisé qui permet l'indexation conjointe de multiples bases de données multimédia aux contenus potentiellement hétérogènes. Nous montrons que cette approche, qui repose sur le calcul décentralisé d'un dictionnaire visuel, fournit des résultats de recherche équivalents aux approches centralisées tout en garantissant un passage à l'échelle aisé.

Mots Clef

Recherche d'images par le contenu distribuée (D-CBIR), indexation multimédia, dictionnaires visuels, clustering non-supervisé, quantification vectorielle (VQ).

Abstract

In order to face the new scalability challenges in content-based indexing raised by the explosion of the available contents amounts, we present a novel decentralized system which allows joint indexing of multiple multimedia databases with potentially heterogeneous contents. We show that this technique, which relies on the decentralized learning of a visual codebook, gives retrieval results which equals centralized approachs, while ensuring effortless scaling.

Keywords

Distributed content-based image retrieval (D-CBIR), multimedia indexing, visual codebooks, unsupervised clustering, vector quantization.

1 Introduction

Les progrès récents en recherche d'information dans les bases de données permettent aujourd'hui de répondre à des requêtes multimédia par leur contenu. Grâce à des processus d'indexation performants, ces systèmes sont capables de renvoyer des réponses pertinentes en un temps raisonnable, dès lors que la base contient au plus quelques millions de documents. La plupart de ces stratégies d'indexation sont basées sur des dictionnaires visuels, qui reproduisent les principes des index textuels, mais à l'aide de mots visuels [20].

Les méthodes actuelles d'indexation multimédia sont conçues pour des environnements centralisés, où l'on fait l'hypothèse que les ressources de stockage des unités de traitement sont sans limite. Or, dans le cas où l'on souhaite traiter des bases au delà du milliard de documents, cette limite est franchie, étant donnée l'immense quantité de métadonnées produite par les processus d'indexation par le contenu. Une solution à ce problème est de considérer des environnements décentralisés. Un système décentralisé permet en effet d'indexer des bases multimédia sans limite théorique, puisque chaque unité peut stocker à la fois les documents et leurs métadonnées. Cependant, il est nécessaire d'adapter les traitements à cet environnement particulier.

Dans cet article, nous portons une attention particulière à l'une des pièces maîtresses : le dictionnaire visuel.

Pour ce faire, nous commençons par présenter l'algorithme K-Means, qui permet le calcul de ce dictionnaire dans un environnement centralisé. Puis, nous présentons les méthodes existantes concernant sa décentralisation. Nous introduisons alors l'approche retenue, qui s'appuie sur un algorithme de K-Means décentralisé basé sur des protocoles "Gossip". Puis, nous proposons une architecture pour l'indexation décentralisée incluant ce processus de calcul du dictionnaire. Enfin, nous présentons une étude expérimentale des paramètres intervenant dans ces procédures, ainsi qu'une comparaison aux systèmes centralisés.

2 Algorithme K-Means

Étant donnés un ensemble $\mathbf{X} = \{x_1, ..., x_n\}$ de n échantillons de $\mathbb{R}^D$ et un nombre K de clusters désirés, l'algorithme K-Means calcule itérativement une partition de l'espace en K régions. Ces régions correspondent à la partition de Voronoi associée à un ensemble $\mathbf{Z} = \{z_1, ..., z_K\}$ de K vecteurs de $\mathbb{R}^D$ nommés centroïdes, que l'algorithme fait évoluer de façon a minimiser progressivement l'erreur quadratique intra-cluster, minimisant ainsi la fonction de coût suivante :

$$J(\mathbf{Z}) = \sum_{i=1}^{n} \min_{k=1..K} \|x_i - z_k\|^2. \quad (1)$$

Un cluster $\mathbf{C}_k$ correspond donc à l'ensemble des vecteurs x_i dont z_k est le plus proche voisin :

$$\mathbf{C}_k = \{x_i \ / \ k = \arg\min_{l=1..K} \|x_i - z_l\|^2\}. \quad (2)$$

Par la suite, tout vecteur entrant $\mathbf{x}$ de $\mathbb{R}^D$ peut être quantifié en un des vecteurs z_k de $\mathbf{Z}$, tel que :

$$q(\mathbf{x}) = \arg\min_{z_k \in \mathbf{Z}} \|\mathbf{x} - z_k\|^2 \quad (3)$$

Afin de trouver une partition qui optimise J, K-Means procède en deux étapes, répétées jusqu'à ce que l'erreur quadratique passe sous un seuil ϵ donné, ou jusqu'à ce qu'aucune amélioration n'ait été apportée entre deux itérations consécutives :

Etape 1 : Assignation des clusters - Chaque échantillon d'apprentissage x_i de $\mathbf{X}$ se voit assigner l'indice k du centroïde z_k le plus proche.

Etape 2 : Mise a jour des centroïdes - Chaque centroïde z_k est recalculé comme le barycentre du cluster $\mathbf{C}_k$ correspondant.

A chaque exécution de l'étape 1, le critère de choix du plus proche voisin garantit qu'on sélectionne toujours le partitionnement de l'ensemble d'apprentissage qui minimise l'erreur quadratique, compte tenu des centroïdes courants. Puis, à chaque exécution de l'étape 2, le critère de choix des barycentres des clusters comme vecteurs référents garantit à son tour une erreur minimale compte tenu de la partition précédemment fixée. Alternant ainsi entre amélioration de la partition et mise à jour des centroïdes, l'algorithme K-Means parvient, à convergence, à un minimum local de J, en fournissant une configuration localement optimale des centroïdes qui minimise la variance intra-cluster [14, 15].
Les centroïdes z_k résultants peuvent être utilisés comme un dictionnaire de K mots qui, en minimisant l'erreur de quantification du modèle (i.e. J), maximise la fidélité des signatures produites.

Améliorations courantes :

- L'initialisation des centroïdes peut se faire aléatoirement, ou par splitting [14].
- L'implémentation stricte (LBG, [14]) de K-Means converge souvent vers des configurations où certains clusters demeurent vides. On peut éviter ces minima non-pertinents par codeword-shifting (e.g. ELBG, [17, 18])

3 Décentralisation : approches existantes

De nombreuses stratégies ont été proposées dans le but de décentraliser l'algorithme K-Means sur plusieurs noeuds interconnectés, possédant chacun une unité de calcul et un jeu de données propres.

Assignation des clusters. La plupart des méthodes tirent profit du parallélisme intrinsèque de l'étape 1 (assignation des clusters). En effet, pour chaque échantillon x_i, trouver le centroïde z_k le plus proche ne nécessite que d'avoir localement accès à l'ensemble Z des centroïdes. Cette étape peut s'effectuer en parallèle pour chaque échantillon. Ainsi, le partitionnement de Voronoi de l'espace global des données ne pose aucun problème de décentralisation pourvu que les systèmes puissent partager des centroïdes commun.

Cependant, les méthodes de l'état de l'art divergent quant à la stratégie de partage des centroïdes entre les noeuds :

Agrégation centralisée. [4, 6] proposent une combinaison centralisée par un noeud maître responsable de rassembler les centroïdes $z_k^{(i)}$ calculés localement par chaque noeud i, les agréger et en redistribuer une unique version. Tout passage à l'échelle est alors limité par le trafic réseau très important généré vers et depuis le noeud maître. [6] limite ce trafic en réalisant la combinaison seulement toutes les τ itérations, mais le système est alors susceptible de converger vers des minima locaux non-désirés ou des états oscillants lorsque la répartition des données est inhomogène.

Agrégation hiérarchique. [16] propose une organisation hiérarchique des agrégations, en suivant un arbre couvrant sur le réseau (par un protocole de type ECHO/PROBE). Une fois que le noeud racine de l'arbre, initiateur du processus, calcule la combinaison finale, le résultat est redistribué en sens inverse. [7] met en évidence que si on gagne bien en trafic, la non-redondance limite la robustesse du système : une branche entière d'informations est perdue dès lors qu'un noeud intermédiaire faillit.

Agrégation par raffinement d'estimées locales. [2, 9, 3] optent pour une agrégation locale prenant en compte la redondance comme un élément vertueux du processus. Ainsi, dans [9], chaque noeud demande à ses voisins leurs centroïdes et les combine à son propre modèle pour améliorer itérativement une approximation locale. Dans le même esprit, [3] réduit l'erreur entre noeuds par échantillonnage aléatoire et fournit asymptotiquement une bonne estimation du modèle global en chaque noeud. Ces approches requièrent néanmoins des échanges requête/réponse permanents, n'offrent pas de garantie formelle de convergence temporelle et spatiale, et [8] met en évidence qu'elles se comportent assez mal lorsque la répartition des données est hétérogène.

4 Epidemic K-Means

L'algorithme Epidemic K-Means (*cf.. Algorithme 2*), proposé par Di Fatta [8, 7] en 2011, apporte une solution décentralisée à la mise à jour des centroïdes (étape 2), en s'appuyant sur un algorithme nommé "Push-Sum" [13] :

1. On calcule localement pour chaque noeud i le vecteur $S^{(i)} = (s_1^{(i)}, n_1^{(i)}, s_2^{(i)}, n_2^{(i)}, ..., s_K^{(i)}, n_K^{(i)})$ composé des sommes partielles et des cardinaux suivants :
$$s_k^{(i)} = \sum_{x_{kj}^{(i)} \in \mathbf{C}_k^{(i)}} x_{kj}^{(i)} \quad \text{et} \quad n_k^{(i)} = \left|\mathbf{C}_k^{(i)}\right| \qquad (4)$$
2. On fait appel à "Push-Sum" pour calculer et répandre de manière décentralisée la somme globale des $S^{(i)}$:
$$s_k = \sum_i s_k^{(i)} \quad \text{et} \quad n_k = \sum_i n_k^{(i)}. \qquad (5)$$
3. On peut alors en déduire localement les centroïdes communs :
$$z_k = s_k / n_k \qquad (6)$$

Algorithme Push-Sum. Push-Sum [13] est un algorithme qui permet de réaliser des sommes vectorielles de manière décentralisée (*cf.. Algorithme 1*). Il entre dans le cadre des protocoles dits "Gossip" ou "épidémiques" [5]. De mise en œuvre très simple, il peut être utilisé de manière asynchrone [1], et offre des garanties formelles de passage à l'échelle et de convergence en temps exponentiellement rapide quel que soit le nombre de noeuds impliqués.
Si chaque noeud i possède une donnée vectorielle initiale $x_0^{(i)}$, Push-Sum fournit itérativement à i un vecteur local $x_t^{(i)}$ qui tend vers la somme globale de tous les x_0 initiaux. Son principe est le suivant :

1. Un poids $w_0^{(i)}$ est associé à chaque vecteur local $x_0^{(i)}$.
2. A chaque passe de Gossip, chaque noeud envoie une copie du couple $(x_t^{(i)}, w_t^{(i)})$ qu'il possède localement. Les données $x_t^{(i)}$ et les poids $w_t^{(i)}$ sont alors divisés par le nombre de redondances induites par cette copie, de manière à respecter une propriété de "conservation de masse". La somme des poids w_t reste alors constante sur le réseau.
3. A réception d'un message, un noeud i ajoute le couple reçu $(x_t^{(j)}, w_t^{(j)})$ à son propre $(x_t^{(i)}, w_t^{(i)})$ pour former $(x_{t+1}^{(i)}, w_{t+1}^{(i)})$.
4. On alterne envoi et réception autant de fois que de passes de Gossip N_{Gossip} souhaitées.

Kempe [13] montre que Push-Sum calcule aussi bien :
- Des moyennes en utilisant des w_0 non nuls
- Des sommes en fixant le poids d'un noeud arbitraire i à 1 et les autres à 0.

L'algorithme s'étend facilement à l'envoi de plusieurs messages par noeud à chaque passe [12]. L'implémentation originale d'Epidemic K-Means [8] utilise une version optimisée de Push-Sum reposant sur des messages symétriques (convergence plus rapide), mais notre étude s'est concentrée sur la version intuitive qui donne malgré tout de très bons résultats de convergence.

Algorithme 1 : PushSum (vectoriel)

Données :
- T_{Gossip} : Nb de passes de Gossip à exécuter (**in**)
- w_0 : Poids initial (**in**)
- $\mathbf{x} = (x_1, ..., x_n)$: Vecteur local à sommer (**in/out**)

début
  $w \leftarrow w_0$
  Choisir aléatoirement un noeud voisin i
  Envoyer $(w, \mathbf{x})$ à i
  pour t ***de*** *1* ***à*** T_{Gossip} **faire**
    Recevoir un message $(w', \mathbf{x}')$
    $w \leftarrow (w + w')/2$
    $\mathbf{x} \leftarrow (\mathbf{x} + \mathbf{x}')/2$
    Choisir aléatoirement un noeud voisin i
    Envoyer $(w, \mathbf{x})$ à i
  fin
  $\mathbf{x} \leftarrow \mathbf{x}/w$
fin

Algorithme 2 : Epidemic K-Means

Pour un noeud i
Entrées :
- $X_i = \{x_i\} \subset X$: jeu de données local
- T : Nb d'itérations de K-Means à exécuter.
- T_{Gossip} : Nb de passes de Gossip pour chaque itération
- K : Nb de clusters souhaités

Sorties :
- Un ensemble de centroïdes locaux $Z_i = \{z_1^i, z_2^i, ..., z_K^i\}$ qui optimisent J sur le jeu complet des données X

début
  Initialiser aléatoirement les z_k^i, $1 \leq k \leq K$
  pour t ***de*** 1 ***à*** T **faire**
    pour tous les $x_i \in X_i$ **faire**
      Trouver le centroïde z_k le plus proche de x_i
      Assigner x_i au cluster $\mathbf{C}_k$
    fin
    pour k ***de*** 1 ***à*** K **faire**
      Calculer la somme locale $s_k^i = \sum_{x_i \in \mathbf{C}_k} x_i$
      Calculer le cardinal $n_k^i = |\mathbf{C}_k^i|$
    fin
    si $i = 1$ **alors** $w_0^i \leftarrow 1$ **sinon** $w_0^i \leftarrow 0$
    PushSum $\left(T_{Gossip}, w_0^i, (s_1^i, n_1^i, ..., s_K^i, n_K^i)\right)$
    pour tous les $k \in \{1, ..., K\}$ **faire**
      $z_k^i \leftarrow s_k^i / n_k^i$
    fin
  fin
fin

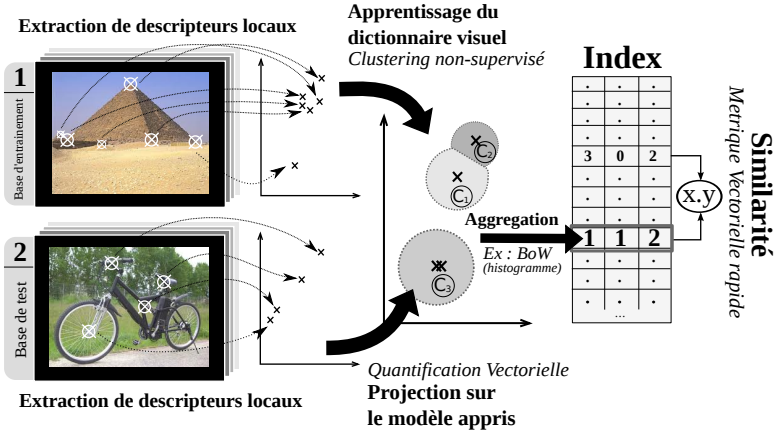

FIGURE 1 – Processus général d'indexation par dictionnaire visuel.

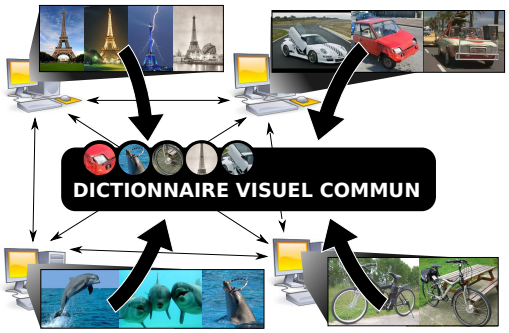

FIGURE 2 – Les unités d'indexation locales, qui possèdent chacune leur base propre, conviennent d'un modèle commun pour indexer conjointement leurs contenus.

5 Système d'indexation décentralisé

Les capacités de clustering distribué offertes par l'algorithme Epidemic K-Means nous ont permis de concevoir un système d'indexation multimédia par le contenu totalement décentralisé. Ce système est capable d'indexer conjointement des bases d'images distinctes, résidant sur différents noeuds dotés de ressources de calcul et de stockage locales.

Procédure d'indexation. La procédure d'indexation que nous proposons est une extension décentralisée de la chaîne d'indexation par dictionnaire visuel classique *(cf. Figure 1)*. En supposant que M noeuds interconnectés en réseau hébergent chacun une base d'images $X^{(i)} = \{x_1^{(i)}, x_2^{(i)}, ..., x_N^{(i)}\}, 1 \leq i \leq M$, le système calcule pour chaque image $x_j^{(i)}$ une signature vectorielle $y_j^{(i)}$, stockée localement sur le noeud qui l'héberge. Cette signature permet de comparer rapidement deux images provenant de n'importe quel noeud i' du réseau grâce à une métrique vectorielle usuelle. Chaque noeud i du système exécute la procédure d'indexation locale suivante :

1. On extrait un ensemble $D_j^{(i)}$ de descripteurs visuels locaux (e.g. SIFT) pour chaque image $x_j^{(i)} \in X^{(i)}$ hébergée par i
2. On calcule de manière décentralisée, à l'aide d'Epidemic K-Means, un modèle global de l'espace de tous les descripteurs extraits par tous les noeuds du réseau, qui partitionne l'espace de description en K régions de Voronoi. Ce modèle constitue alors un dictionnaire visuel $Z = \{z_1, ..., z_K\}$ commun à tous les noeuds, qui rend compte de l'ensemble des images présentes dans le système.
3. On projette par quantification vectorielle les descripteurs $D_j^{(i)}$ de chaque image sur le dictionnaire visuel commun Z ainsi obtenu.
4. On utilise une méthode de la littérature pour agréger ces projections en des signatures $y_j^{(i)}$: histogramme d'occurrences pour le cas BoW [20], déviation au premier ordre pour VLAD [11], au second ordre pour VLAT [19], etc. Toute signature qui s'appuie sur un dictionnaire visuel peut être théoriquement utilisée.

Avantages. Le système d'indexation décentralisé que nous proposons permet théoriquement de traiter des bases sans limite de taille, sous réserve que l'on puisse ajouter sans fin de nouvelles unités de traitement. En effet, le principal facteur limitant est la capacité à transférer des données d'une unité à l'autre, via un réseau ou un bus mémoire, suivant la nature des unités de traitement.

Or, les seules informations qui transitent entre les unités sont les messages de l'Epidemic K-Means, dont la taille est très faible devant la celle des métadonnées que nous aurions à transférer dans un contexte centralisé. Comme nous le détaillons dans la section 6, il est aussi possible de jouer sur la taille et le nombre de messages pour mieux s'adapter à la nature du réseau considéré.

Par conséquent, il est possible d'utiliser les protocoles d'indexation les plus aboutis, où l'on extrait des quantités très élevées de descripteurs visuels. Par exemple, en indexation vidéo, il devient de plus en plus courant d'extraire des millions de descripteurs visuels par extrait vidéo [21]. Ainsi, si on souhaite utiliser ce protocole d'extraction sur des millions voir des milliards d'extraits vidéo, les environnements centralisés sont dépassés :

– Soit on distribue le calcul des descripteurs, mais il faut alors tout transférer sur une même machine.
– Soit on évite les transferts en calculant tout sur une même unité de calcul très puissante, mais qui sera forcément dépassé tôt ou tard par le nombre.

En revanche, avec notre système d'indexation décentralisé, il suffit d'ajouter de nouvelles unités de calculs au réseau pour traiter des bases toujours plus grandes.

FIGURE 3 – Images de la base Holidays (Inria)

6 Expériences et résultats

Afin d'évaluer les performances de notre système sur de grandes topologies de réseaux (jusqu'à 1000 noeuds), nous avons mis en place une plateforme de simulation logicielle qui permet d'exécuter un nombre arbitraire de noeuds virtuels sur un ensemble de machines matérielles. Les simulations ont été réalisées avec puis sans synchronisation d'exécution des noeuds.

Le système a été entraîné sur la base d'images Holidays [10], constitué de 1491 documents répartis en 500 requêtes *(cf. Figure 3)*. L'objectif sur cette base est la recherche de quasi-copies (i.e une même scène dans des conditions de prise de vue différente : angle, éclairage, occultations, etc...). Chaque image est décrite par un sac de descripteurs SIFT (quelques milliers par image). Les images ont été dispersées sur les noeuds en proportions égales, aléatoirement puis par groupe de requêtes, afin de tester la résistance de notre système à l'hétérogénéité de la distribution.

Nous avons finalement évalué la capacité du système à apprendre de manière décentralisée des dictionnaires locaux cohérents, puis les performances d'indexation de ces dictionnaires en terme de precision moyenne de recherche (mAP) sur la totalité de la base Holidays.

6.1 Apprentissage de dictionnaires visuels

Nous nous sommes en premier lieu assurés que notre système parvenait bien à converger, pour chaque itération de K-Means, vers des agrégats communs $s_k = \sum s_k^{(i)}$ et $n_k = \sum n_k^{(i)}$, quelque soit le nombre de clusters désires K, le nombre de noeuds impliqués M, et le nombre total d'images N introduits dans le système global. Les résultats obtenus sont présentés dans la Figure 4. Nos expériences attestent que la convergence en temps du calcul de dictionnaire est exponentiellement rapide et ne dépend quasiment pas du nombre M de noeuds en jeu.

Complexité calculatoire. Le nombre d'images N n'a d'influence que sur les étapes locales d'assignation des clusters et de sommation partielle, qui deviennent plus coûteuses (ce qui est déjà le cas en contexte centralisé). Néanmoins, la répartition du jeu de données sur M noeuds divise par M le nombre moyen d'images affectées à un noeud donné, multipliant par M la vitesse d'exécution de ces étapes totalement parallélisées et indépendantes. L'étape de Gossip travaillant au niveau des "pré-centroïdes" $(s_k^{(i)}, n_k^{(i)})$, elle n'est pas impactée par les variations de N, et le surcoût calculatoire qu'elle apporte est totalement

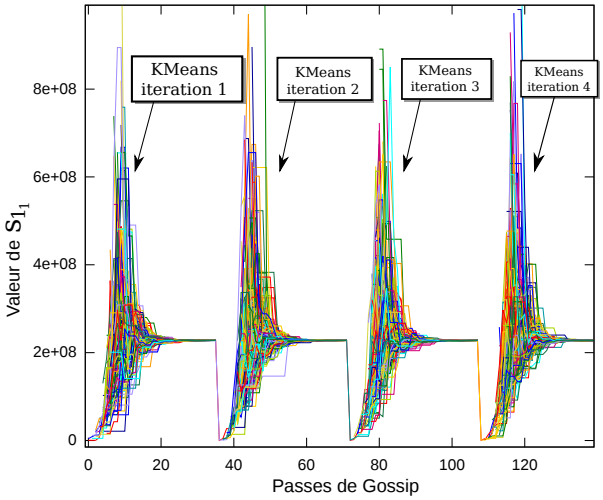

FIGURE 4 – Grâce à Push-Sum, les sommes partielles $s_k^{(i)}$ de tous les noeuds convergent vers leur somme globale s_k en temps exponentiellement rapide, garantissant une cohérence totale des dictionnaires construits (ici, seule $s_{1_1}^{(i)}$ est représentée).

négligeable. Il s'agit en effet d'une simple réception de message, une addition vectorielle, et un envoi de message (la construction et le décodage des messages ne coûte pas de temps de calcul car ils ne contiennent qu'une concaténation des $(s_k^{(i)}, n_k^{(i)})$ et d'un $w^{(i)}$.

Trafic réseau. Un message étant composé des $(s_k^{(i)}, n_k^{(i)})$ et du $w^{(i)}$ concaténés $(1 \leq k \leq K)$, la dimension d'un message est de $(D+1)K+1$ flottants, soit environ 32Ko dans le cas d'un dictionnaire de 64 clusters sur des descripteurs SIFT ($K = 64, D = 128$). A chaque passe de Gossip, M messages sont envoyés (1 message par noeud) et transitent sur le réseau en direction de noeuds aléatoires. Un trafic global de $(M \times D \times K)$ est donc généré sur le réseau à chaque passe. Pour des SIFT et un dictionnaire de 64 clusters, dans le cas où 1000 noeuds sont en jeu, 32Mo transitent sur le réseau à chaque passe, ce qui peut poser des problèmes de débit selon l'architecture réseau sous-jacente.

Cohérence des dictionnaires locaux. Comme nous avons pu le remarquer précédemment, les bonnes propriétés de convergence de Push-Sum amènent les noeuds à converger très rapidement vers un agrégat commun, et donc vers des dictionnaires locaux équivalents. En une trentaine de passes maximum, la cohérence des dictionnaires locaux est assurée, ce qui reste assez négligeable en temps devant les calculs locaux [1]. L'hétérogénéité de la distribution des images sur les noeuds n'impacte ni le temps de calcul ni la cohérence des dictionnaires construits.
Nous pouvons conclure que pourvu que l'on se donne un nombre de passes suffisant pour effectuer l'agrégation par

1. Chaque noeud possède au moins quelques milliers d'échantillons à indexer, une image décrite par des SIFT présentant couramment plusieurs milliers d'échantillons à elle seule

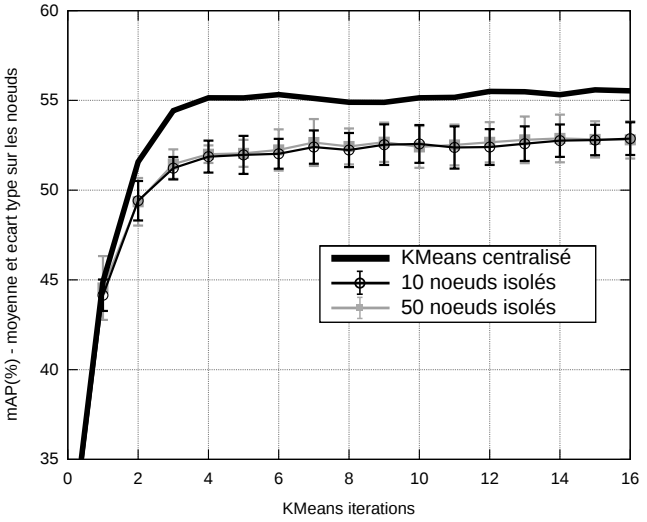

FIGURE 5 – Des dictionnaires locaux entraînés isolément donnent de mauvais résultats (mAP) vis-à-vis d'une approche centralisé (ici $K = 64$).

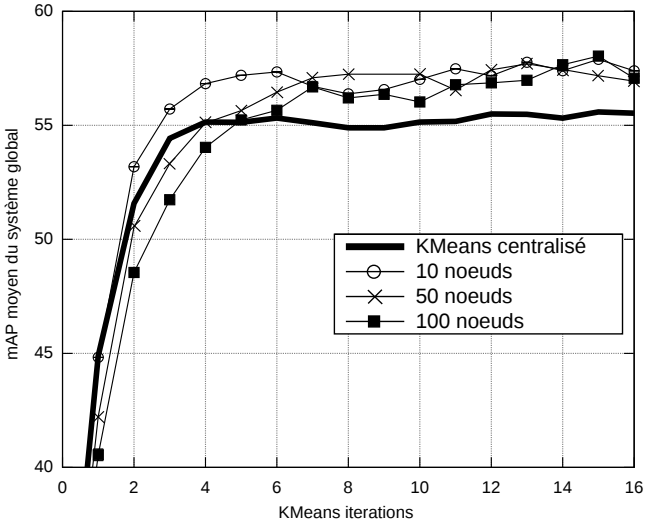

FIGURE 6 – Le dictionnaire commun construit par notre système fournit des résultats de recherche (mAP) meilleurs qu'une solution centralisée (ici $K = 64$).

	K = 4	**K = 8**	**K = 16**	**K = 32**	**K = 64**
référence	42,7	46,5	49,8	53,3	55,1
10 nœuds	44,2	47,9	51,6	54,2	57,0
50 nœuds	42,7	45,9	50,9	55,7	57,2
100 nœuds	44,7	47,3	51,8	53,4	56,0

TABLE 1 – Performances (% mAP) du système pour différentes tailles de dictionnaire K et nombre de noeuds M après 20 itérations.

Gossip des centroïdes ($T_{Gossip} \geq 30$), le caractère décentralisé du système devient totalement transparent du point de vue des dictionnaires construits. Epidemic K-Means est alors équivalent à un K-Means centralisé entraîné sur l'ensemble de la base. Les avantages en terme de temps de calcul et d'empreinte mémoire sont clairs et permettront d'indexer dans la suite de nos travaux des bases de très grandes taille à l'aide de noeuds aux performances de calcul et de stockage raisonnables.

6.2 Performances en indexation

Chaque dictionnaire local généré lors de nos simulations a fait l'objet d'une évaluation de ses performances lorsqu'il est utilisé dans une chaîne d'indexation complète. Nous avons ainsi utilisé ces dictionnaires pour générer des signatures VLAD [11]. La signature VLAD d'une image est un vecteur $\mathbf{v} = (v_1, v_2, ..., v_K)$ de dimension $K.D$ avec

$$v_k = \sum_{x' \in \mathbf{C}_k} (x' - z_k). \quad (7)$$

où $\mathbf{C}_k$ désigne le cluster de centroïde z_k. Cette signature est connue pour ne nécessiter que des dictionnaires de petite taille (de 64 à 1024 clusters). Les performances de nos dictionnaires sont évaluées en terme de mAP (précision moyenne) sur l'ensemble de la base Holidays, par rapport à un dictionnaire de référence de 64 clusters calculé de manière centralisée avec un algorithme de type ELBG, qui réalise une mAP d'environ 55%.

Performances avec convergence de l'étape Gossip. Comme mentionné plus haut, lorsque $T_{Gossip} \geq 30$, les dictionnaires générés ont une totale cohérence et sont ainsi sensiblement égaux aux résultats d'un K-Means centralisé, sous réserve que chaque noeud puisse accéder à un nombre de voisins suffisant [2].

Il s'ensuit que leurs resultats en indexation sont équivalents voire supérieurs à ceux du dictionnaire de référence, tel que le montre la Figure 6. De telles performances signifient qu'un noeud qui ne dispose localement que d'informations parcellaires sur le jeu de données complet est capable de rendre compte de l'espace engendré par l'ensemble des données. Il apprend ainsi à modéliser la totalité de la base (notamment des concepts visuels dont il ne disposait par forcément localement), là où des systèmes entrainés isolément montrent une mauvaise généralisation comme l'atteste la Figure 5.

Les expériences menées pour différents nombres de noeuds M et tailles de dictionnaire K, dont les résultats sont rassemblés en Table 1, montrent que le système décentralisé maintient des performances supérieures à la méthode de référence et présente une mAP très correcte pour de petits dictionnaires.

Performances sous convergence partielle de l'étape Gossip. Nous nous sommes par ailleurs intéressés aux

2. Les expériences réalisées sur des topologies de réseaux telles que des anneaux ou des grilles ont montré que les garanties de convergence en temps étaient alors perdues et que le nombre de phases de Gossip requises pour une parfaite cohérence devenait dépendant du nombre de noeuds

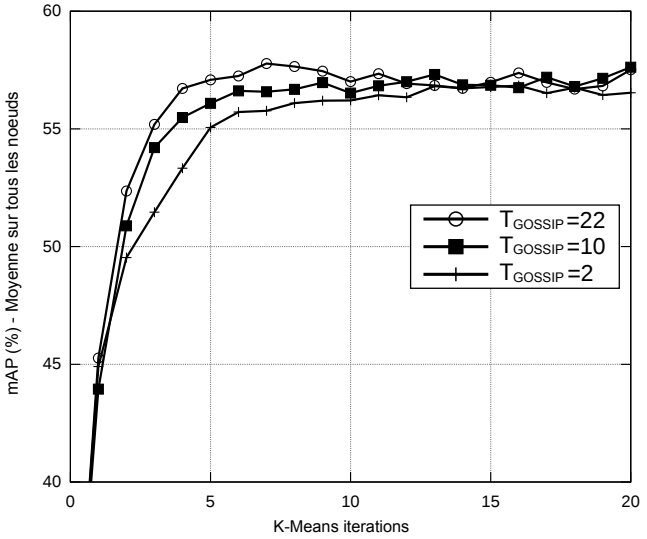

FIGURE 7 – Une faible valeur de T_{Gossip} ralentit la convergence à court terme mais n'impacte pas le comportement asymptotique du système (ici $M = 50$).

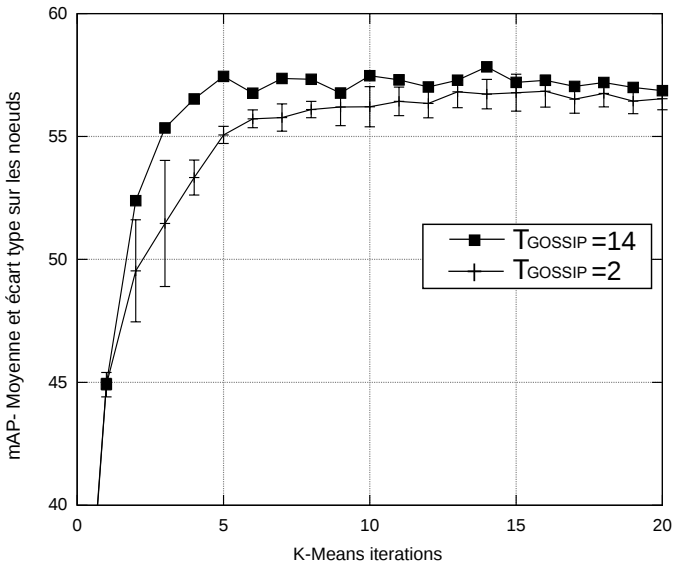

FIGURE 8 – Un T_{Gossip} faible accroît légèrement la variance de la mAP entre noeuds (ici $M = 50$).

performances de nos dictionnaires dans le cas où le nombre de passes de Gossip n'est pas suffisant pour que les noeuds puissent totalement converger vers des centroïdes communs à chaque itération de K-Means. Nos simulations menées sur des réseaux de 50 noeuds pour 64 clusters désirés ont révélé que bien que la stabilisation à court terme des dictionnaires soit ralentie du fait de la convergence partielle de l'agrégation Gossip, le système global maintient malgré tout des performances de recherche asymptotiques équivalentes.

La Figure 7 montre notamment qu'une agrégation en seulement 2 phases de Gossip s'avère sans impact sur la mAP moyenne des noeuds à partir d'une dizaine d'itérations. Il faut cependant noter que la dégradation de la cohérence des dictionnaires naturellement provoquée par un T_{Gossip} faible se traduit par une variance plus élevée des performances de recherche entre noeuds, visible dans la Figure 8.

7 Conclusion

Nous avons proposé un système d'indexation multimédia par le contenu qui s'appuie sur l'apprentissage décentralisé de dictionnaires visuels. Cette approche autorise l'indexation d'une quantité de contenu théoriquement illimitée, par la mise en collaboration de multiples systèmes d'indexation interconnectés.

Appliqué à des bases d'images réparties, ses performances se sont révélées équivalentes à un système centralisé entraîné sur l'ensemble des données. En quelques itérations, les noeuds convergent vers un dictionnaire commun qui offre des résultats de recherche au niveau de l'état de l'art. Les performances sont homogènes à travers le réseau et la vitesse de convergence ne dépend pas du nombre de noeuds en jeu.

D'implémentation simple et légère, notre méthode permet à des systèmes interconnectés de rendre compte de concepts visuels auxquels ils n'ont pas localement accès, là où des systèmes isolés donnent de mauvais résultats.

Les méthodes d'indexation centralisées les plus compétitives font appel à des techniques additionnelles de compression (PCA), de normalisation, etc... qui requièrent elles-aussi une agrégation sur l'ensemble des données. Leur décentralisation fait ainsi l'objet de nos travaux actuels.

Remerciements

Ces travaux sont financés dans le cadre du projet FSN Culture 3D Clouds.

Références

[1] A gossip-based asynchronous aggregation protocol for p2p systems. *Local Computer Networks, Annual IEEE Conference on*, 0 :248–251, 2010.

[2] Giuseppe Campobello, Mirko Mantineo, Giuseppe Patanè, and Marco Russo. Lbgs : a smart approach for very large data sets vector quantization. *Sig. Proc. : Image Comm.*, 20(1) :91–114, 2005.

[3] Souptik Datta, Chris Giannella, and Hillol Kargupta. K-means clustering over a large, dynamic network. In Ghosh et al. [9].

[4] Souptik Datta, Chris Giannella, and Hillol Kargupta. Approximate distributed k-means clustering over a peer-to-peer network. *IEEE Trans. on Knowl. and Data Eng.*, 21(10) :1372–1388, October 2009.

[5] Alan Demers, Dan Greene, Carl Houser, Wes Irish, John Larson, Scott Shenker, Howard Sturgis, Dan Swinehart, and Doug Terry. Epidemic algorithms for replicated database maintenance. *SIGOPS Oper. Syst. Rev.*, 22(1) :8–32, January 1988.

[6] Matthieu Durut, Benoît Patra, and Fabrice Rossi. A discussion on parallelization schemes for stochastic vector quantization algorithms. *CoRR*, abs/1205.2282, 2012.

[7] Giuseppe Di Fatta, Francesco Blasa, Simone Cafiero, and Giancarlo Fortino. Epidemic k-means clustering. In *2011 IEEE 11th International Conference on Data Mining Workshops*, pages 151–158. IEEE, December 2011. Print ISBN : 9781467300056 Issue Date : 11-11 Dec. 2011 On page(s) : 151 - 158.

[8] Giuseppe Di Fatta, Francesco Blasa, Simone Cafiero, and Giancarlo Fortino. Fault tolerant decentralised k-means clustering for asynchronous large-scale networks. *Journal of Parallel and Distributed Computing*, September 2012.

[9] Joydeep Ghosh, Diane Lambert, David B. Skillicorn, and Jaideep Srivastava, editors. *Proceedings of the Sixth SIAM International Conference on Data Mining, April 20-22, 2006, Bethesda, MD, USA*. SIAM, 2006.

[10] Hervé Jégou, Matthijs Douze, and Cordelia Schmid. Hamming embedding and weak geometric consistency for large scale image search. In Andrew Zisserman David Forsyth, Philip Torr, editor, *European Conference on Computer Vision*, volume I of *LNCS*, pages 304–317. Springer, oct 2008.

[11] Hervé Jégou, Florent Perronnin, Matthijs Douze, Jorge Sánchez, Patrick Pérez, and Cordelia Schmid. Aggregating local image descriptors into compact codes. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 1 :3304–3311, 2012. QUAERO.

[12] Márk Jelasity, Alberto Montresor, and Ozalp Babaoglu. Gossip-based aggregation in large dynamic networks. *ACM Trans. Comput. Syst.*, 23(3) :219–252, August 2005.

[13] David Kempe, Alin Dobra, and Johannes Gehrke. Gossip-based computation of aggregate information. In *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science*, FOCS '03, pages 482–, Washington, DC, USA, 2003. IEEE Computer Society.

[14] Y. Linde, A. Buzo, and R.M. Gray. An algorithm for vector quantizer design. *IEEE Transaction on Communication*, 28 :84–94, 1980.

[15] Stuart P. Lloyd. Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 28 :129–137, 1982.

[16] W. T. müller, M. eisenhardt, and A. henrich. Efficient content-based P2P image retrieval using peer content descriptions. In *Internet Imaging V. Edited by Santini, Simone ; Schettini, Raimondo. Proceedings of the SPIE, Volume 5304, pp. 57-68 (2003).*, pages 57–68, December 2003.

[17] Giuseppe Patanè and Marco Russo. Elbg implementation. *International Journal of Knowledge based Intelligent Engineering Systems*, 2 :2–4, 2000.

[18] G. Patané and M. Russo. The enhanced lbg algorithm. *Neural Netw*, 14(9) :1219–37, 2001.

[19] David Picard and Philippe-Henri Gosselin. Improving image similarity with vectors of locally aggregated tensors. In *IEEE International Conference on Image Processing*, Brussels, Belgique, September 2011.

[20] J. Sivic and A. Zisserman. Video google : A text retrieval approach to object matching in videos. In *IEEE International Conference on Computer Vision*, volume 2, pages 1470–1477, 2003.

[21] Heng Wang, Alexander Kläser, Cordelia Schmid, and Liu Cheng-Lin. Action recognition by dense trajectories. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2011.