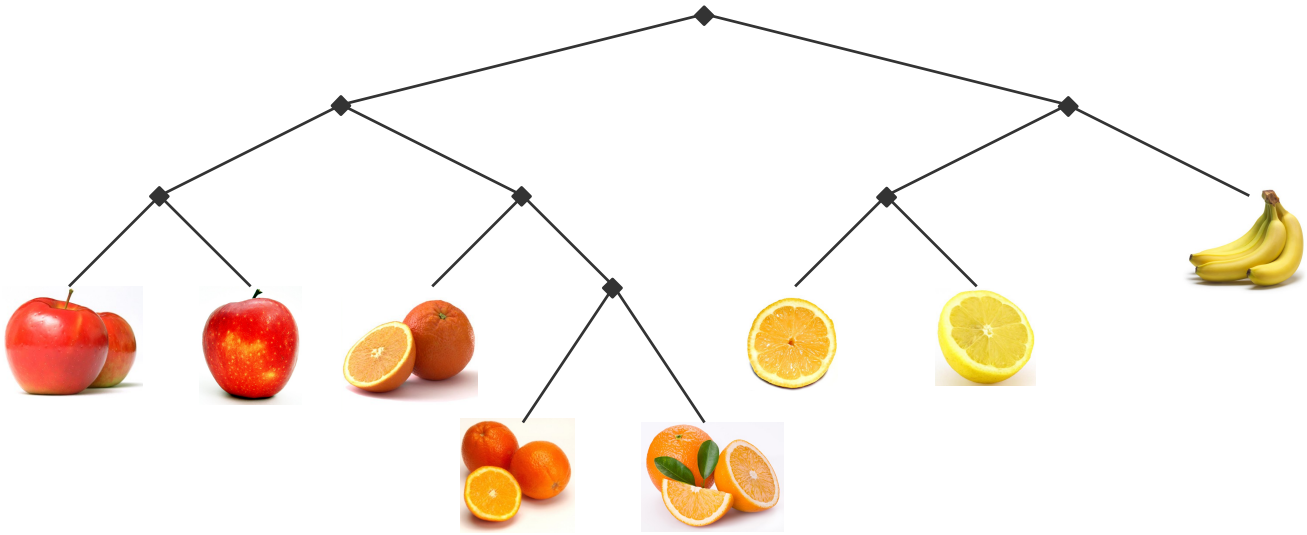
SCALABLE CLUSTERING FOR HIERARCHICAL CONTENT-BASED BROWSING OF LARGE-SCALE IMAGE COLLECTIONS

CHRISTOPHER TIM ALTHOFF



Computer Science Department
University of Kaiserslautern
Germany

September 2010

Bachelor-Thesis
Department of Computer Science
University of Kaiserslautern
P.O. Box 3049
67653 Kaiserslautern
Germany

SUPERVISORS:
Prof. Dr. Prof. h.c. Andreas Dengel
Dr. Adrian Ulges

ABSTRACT

In recent years the explosive growth of digitally stored image and video data has increased the need for tools to search and organize visual data automatically by their content. Browsing environments, which help users to find the images or videos they need by structuring image and video collections, are one solution to this problem. Therefore, image clustering techniques are needed that group semantically related images, are highly scalable, and produce balanced structures to be useful for content-based browsing.

In this thesis both flat and hierarchical `k-means` clustering techniques are compared to hierarchical agglomerative ones particularly with respect to scalability and balancing requirements, using multiple standard features and real-world datasets. Another main contribution is a simple and efficient strategy to enforce a more balanced clustering based on a hierarchical frequency sensitive variant of the online `k-means` algorithm, which we call `hbo-k-means`. This method will be integrated in a practically employed system within the FIVES Project.

ZUSAMMENFASSUNG

Das exponentielle Wachstum digital gespeicherter Bild- und Videodaten in den letzten Jahren hat einen Bedarf an Werkzeugen geschaffen, Daten inhaltsbasiert zu durchsuchen und organisieren. Browsing-Umgebungen, die Nutzern bei der Suche nach Bildern und Videos durch die Strukturierung der Daten unterstützen, stellen eine Lösung dieses Problems dar. Aus diesem Grund werden Bild-Clustering-Techniken benötigt, die in der Lage sind semantisch verwandte Bilder zu gruppieren, hoch skalierbar sind und balancierte Strukturen erzeugen, um für inhaltsbasiertes Durchsuchen von Nutzen zu sein.

Als einen ersten Schritt in diese Richtung werden in dieser Arbeit flache und hierarchische `k-means`-Methoden mit hierarchisch agglomerativen Clustering-Verfahren mit Schwerpunkten auf Skalierbarkeit und Balanciertheit verglichen. Dabei werden mehrere Standard-Features und anwendungsnahe Datensätze verwendet. Ein weiterer Beitrag dieser Arbeit ist eine einfache und effiziente Strategie, genannt `hbo-k-means`, um balancierte Clusterings zu erzwingen. Die Methode basiert auf einer häufigkeitssensitiven Variante des online `k-means` Algorithmus und wird in ein praktisch eingesetztes System im Rahmen des FIVES-Projektes integriert.

We have seen that computer programming is an art,
because it applies accumulated knowledge to the world,
because it requires skill and ingenuity, and especially
because it produces objects of beauty.

— Donald E. Knuth [38]

ACKNOWLEDGMENTS

This thesis originates from my work as a student research assistant at the German Research Center for Artificial Intelligence (DFKI), where I have been a member of the Multimedia Analysis and Data Mining research group since May 2009. I would like to thank Prof. Dr. Prof. h.c. Andreas Dengel for the interesting possibilities at DFKI.

Especially, I am heartily thankful to my supervisor, Dr. Adrian Ulges, whose encouragement, guidance and support from the initial to the final level enabled me to develop an understanding of the subject.

I offer my regards and blessings to all of those who supported me in any respect during the completion of the project. Particularly, I feel very grateful to Damian Borth for the helpful tips and discussions and to Klaus-Dieter Althoff, Michael Holzhauser and Anna Scheer for proof-reading the manuscript.

Lastly, my deepest gratitude goes to my family, and especially my parents, for all the possibilities they have created for me and for giving me the best possible support and personal reflection.

Christopher Tim Althoff

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

1 INTRODUCTION

Over the last few years digital data has been growing exponentially. The main reason is the vast increase of sharing image and video data over the Internet. Here are a few key statistics to illustrate this development.

YouTube is a video-sharing website on which users can upload, share, and view videos. Alexa ranks YouTube as the third most visited website on the Internet, behind Google and Facebook (see [33]). Youtube exceeds 2 billion views a day, which is nearly double the prime-time audience of all three major U.S. broadcast networks combined. Additionally, 24 hours of video are uploaded every minute. In other words, more video is uploaded to YouTube in 60 days than all three major US networks created in 60 years. Also interesting is that the combination of better search and discovery (in addition to more content) has driven the daily time each user spends on YouTube on average up 55% in 2009 (see [44]; Effective May 2010).

One of the most popular image hosting websites is Flickr. Flickr hosts more than 4 billion images. The social networking website Facebook even registers around 2.5 billion uploads to the site each month (see [22]; Effective January 2010). Therefore, it is a little less surprising that the number of digital cameras and camera phones in the world surpassed 1 billion in 2007 (see [26]; Effective March 2008).

Where does all this data come from? As already indicated online portals like YouTube, Flickr or Facebook offer data in large amounts. This content is mainly generated by users sharing their photos and videos. It has become easy to acquire huge volumes of digital data because digital acquisition devices (still image and video cameras) are already wide-spread. But huge image collections are also becoming common due to commercial efforts like *google street view* (see [55]) or broadcasting networks, the archives of which contain increasingly more digital content (see [8, 14]). This creates the need for efficient ways of searching through the content, and imposes the challenge to organize and structure it, so that users can browse and find the images and videos they are looking for.

Different Approaches On Browsing Multimedia Databases

Currently, search for specific content is mostly done through a *query-by-text* approach, for which images and videos has to be manually annotated. Borth et al. state a few drawbacks. Manual annotation is a very time-consuming process which might lead to subjective results depending on the person doing the annotation. Furthermore, only content that has been annotated can be retrieved afterwards and the quality of

search results highly depends on the quality of the annotation. In online-portals like *youtube.com* and *flickr.com* the owners of the uploaded data provide the meta-data ("tags") themselves, which enables them to manipulate the availability of their content for a wide range of possible queries. Consequently more suited content might be suppressed which reduces the quality of the entire search result (see [8]).

Another approach more suited for large multimedia databases is to describe objects by their content or semantic information. The semantic information of an image is what can be inferred after seeing the image, for example the presence or absence of specific objects, their attributes or relative positions. Understanding the semantics of an image has uncountable applications ranging from object recognition, face detection to image retrieval (see [1]). The problem of formulating proper queries led to the definition of the *semantic gap*, "a lack of coincidence between the information that one can extract from the data and the interpretation of the same data for a user in a given situation." (see [51]). Multiple approaches try to bridge this semantic gap, often leading to user interface design (see [8]).

A different content-based strategy is the *query-by-example* approach, where an image or video is given to the system to represent the query, bypasses the inherent problem of query formulation in this space (see [8]). However, while so far most approaches of image database management have focused on the query-by-example approach, the effectiveness remains questionable. Obviously, it is often difficult to find a suitable query image. In addition, repetitive queries tend to become trapped among a small group of undesired images, providing only a localized view on the database (see [11]).

Content-Based Browsing Environments

A third approach is content-based browsing environments, which offer an alternative to conventional search-by-query approaches. They try to organize and structure the given collection so that users can find the entity they need. Instead of presenting a localized view of the database, browsing environments also offer an overview of the entire database. Also, they allow the user to dynamically redefine their search, which is convenient if the user does not know yet what exactly they he or she is looking for.

Examples for content-based browsing systems are RotorBrowser [18] or VideoSOM [6]. The concept of similarity pyramids for browsing was introduced in [11]. This concept was also applied to video databases in a system called ViBE [12]. Another system for similarity-based browsing of multimedia databases, *Navidgator*[1], was introduced by Borth et al.[2] in [8]. Navidgator serves as a basis for this thesis.

1 `http://madm.dfki.de/navidgator-howto/navidgator-howto.html`

2 Multimedia Analysis and Data Mining research group of the German Research Center of Artificial Intelligence

How do the browsing environments organize a given collection? What underlies systems like Navidgator is a (hierarchical) clustering. This process refers to the unsupervised classification of patterns (observations, data items, or feature vectors) into groups, so called clusters, which is called *clustering* (see [34]). Clustering algorithms are used to build the similarity based structure upon the image database, on which browsing environments rely. Based on the hypothesis that semantically similar images tend to be clustered into the same groups, clustering can both improve the result quality and speed up the retrieval itself, if the search space is restricted to specific clusters of semantically related images (see [13, 36]). Figure 1 illustrates how the hierarchical clustering is used to produce a tree representation used in browsing environments. Part (a) shows images represented by their feature vectors in some feature space. These are then clustered hierarchically in (b) leading to the tree representation in (c).

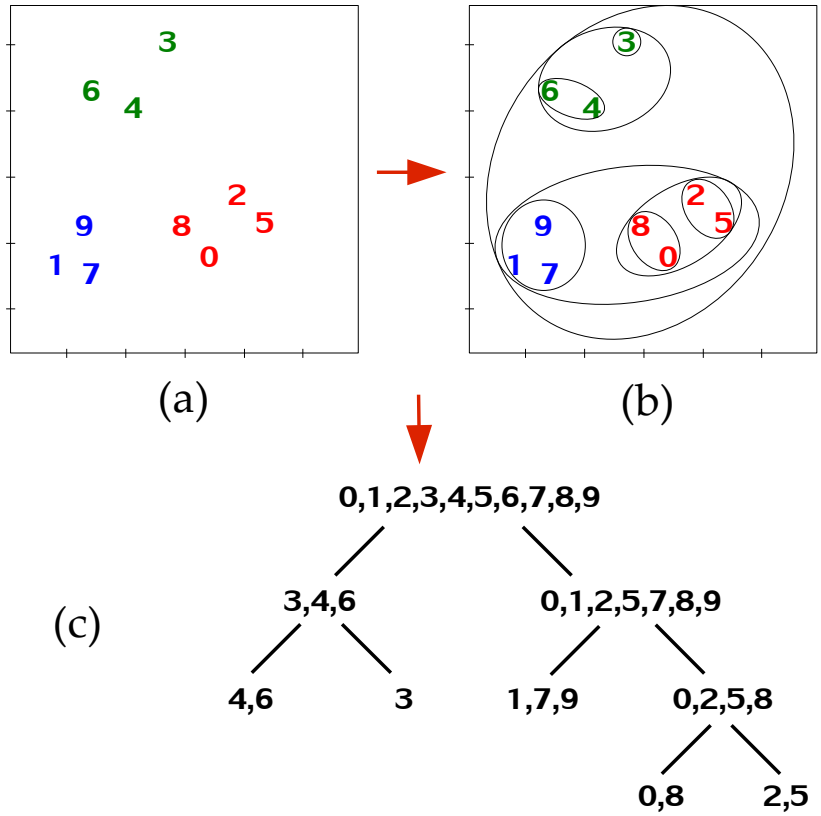


Figure 1: An illustration of the general approach of hierarchical clustering. Feature vectors in some feature space (a) are clustered hierarchically (b), which is then transformed into a tree representation (c).

Systems like Navidgator enable the user to easily and efficiently browse a database via a graphical user interface. The Navidgator interface is shown in Figure 2. The Figure also illustrates the relationship between the interface and the underlying clustering tree, which is the result of a hierarchical clustering algorithm. First, the

user has to define a focus image which basically determines the direction in which the clustering tree will be explored (a). The user is able to redefine the focus image at any point during browsing. The exploration of the database is performed by the given zooming tools (b). The user can either *zoom-out* or *zoom-in* in the database. A *zoom-in* action will narrow down the available images according to his query image (e,f), and a *zoom-out* action will display a coarser level of the database to the user (g,h). The depth of the database and the users current position are visualized by a vertical bar next to the zooming tools enabling an intuitive orientation.

Challenge: Scalability

A key problem with Navidgator is that it uses a hierarchical agglomerative bottom-up clustering. This method works by successively merging clusters together depending on their pairwise distances. Unfortunately, it does not scale very well due to the required distance matrix of size $O(N^2)$, where N is the number of images. In practice, only collections of up to 10.000 images could be clustered and browsed. In many practical applications, however, there is a need for highly scalable image clustering algorithms, which are able to process up to 1 million images using a reasonable amount of resources.

Challenge: Balancing

For image browsing environments clustering techniques are needed, which are not only able to group semantically related images, but will produce balanced structures to be useful for the described task. These balanced structures are more likely to represent the semantic complexity of the dataset in a comprehensible way, because they try to equally distribute the semantic complexity on each single level. In contrast, unbalanced structures lack regularity on which the user orientation implicitly relies. The balancing requirement will be illustrated in more detail in section 2.3.

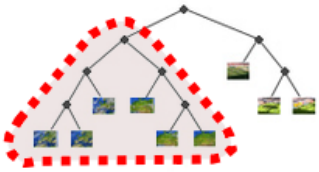
In this thesis the clustering back-end of Navidgator is evolved with respect to both scalability and balancing. Since this is a problem of the underlying hierarchical agglomerative approach, other methods need to be examined. Therefore, multiple variants of the popular `k-means` algorithm are compared to hierarchical agglomerative clustering techniques. `k-means` is linear in the number of images and therefore scales very well. A second main contribution is a simple and efficient strategy to enforce a more balanced clustering based on a frequency sensitive variant of the online `k-means` algorithm, which we call `hbo-k-means`. This clustering technique will be integrated in Navidgator. Particularly, the resulting extended system will be contributed to the FIVES project (Forensic Image and Video Examination Support[3]). FIVES brings together partners from law enforcement, academia and industry to work together to

3 `www.fives.kau.se`

(a) The big image displayed at the top is called the "focus image". It defines what kind of pictures you are currently interested in.

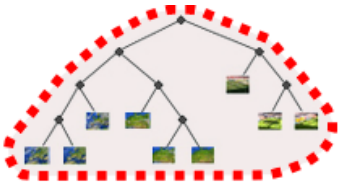
(b) You can change your zoom factor with the controls beneath the focus image.

(c) The images in a database view...



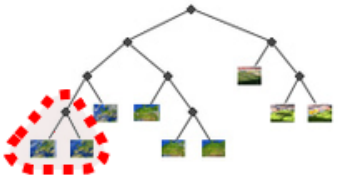
(d) ...can be found in a certain subtree of our browsing structure.

(e) Let us first zoom out such that ...



(f) ...the database view shows samples from the whole collection.

(g) We now zoom into the collection, and the images in the database view get fewer and fewer...



(h) ...and are more and more similar to the focus image.

Figure 2: The Navidgator browsing interface and its relationship to the underlying clustering tree. Also, zoom-in and zoom-out operations are illustrated (images from `http://madm.dfki.de/navidgator-howto/navidgator-howto.html`)

enhance the state-of-the-art for tools available to law enforcement in the investigation of child sexual abuse. The contributed system will be used by law enforcement officers to structure and browse confiscated materials.

Outline

The thesis is organized as follows. Chapter 2 describes the problem of image clustering and gives a brief overview of state-of-the-art solutions. The focus is on clustering algorithms, validity indices to measure the quality of clusterings and different approaches to enforce balanced clustering results. In Chapter 3 deals with the application of clustering techniques for image browsing, describing the particular requirements and our proposed variant of the `k-means` algorithm `hbo-k-means`. Chapter 4 provides the experimental results. Multiple clustering techniques are examined using various standard features on a number of real-world datasets. Special attention is paid to the impact of hierarchical clustering and the enforcement of balancing. Finally, our proposed method is demonstrated to be scalable on a large-scale dataset of youtube keyframes. The thesis is concluded in Chapter 5 with a summary of the results and contributions and a discussion of unsolved problems and future work.

2 STATE-OF-THE-ART

This chapter gives a brief introduction to clustering and a summary of current approaches. First, clustering is defined and the basic approaches are introduced. We then take a look at how to measure clustering quality. Finally, different approaches to enforce a balanced clustering are described.

2.1 CLUSTERING

Clustering refers to the unsupervised classification of patterns (observations, data items, feature vectors, or in our case images/videos) into groups, so-called clusters, such that patterns within the same cluster are similar, and those assigned to different clusters are not (see [34]). Clustering is a subject of research in many areas such as statistics, pattern recognition, machine learning and data mining. A wide variety of clustering algorithms has been proposed for several applications, including image segmentation, object and character recognition, document retrieval, and data mining (see [34, 39]).

Clustering techniques broadly fall into one of the following groups (see [34, 39]):

- *Partitional Clustering*: These algorithms directly decompose the data into a set of disjoint clusters. They attempt to determine partitions that optimize a specific criterion function (optimization is often done iteratively).
- *Hierarchical Clustering*: By refining clusters recursively, these algorithms produce a nested tree of partitions. These algorithms successively merge smaller clusters or split larger ones (see agglomerative vs. divisive below).
- *Density-based Clustering*: The key point of these algorithms is to create clusters based on density function. The main advantage of this class is to create arbitrarily shaped clusters.
- *Grid-based Clustering*: These algorithms quantize the search space into a finite number of cells and have been mainly proposed for spatial data mining.

This thesis focuses on hierarchical clustering. Here, another common distinction is made between *agglomerative* and *divisive* methods, an aspect of algorithmic structure and operation. Agglomerative approaches start with singleton clusters, which contain only one sample, and successively merge clusters together until a stopping criterion is satisfied. These methods are also referred to as *bottom-up*. In contrast, a divisive

method starts with all patterns in one single cluster and performs splitting operations until again a stopping criterion is satisfied. Obviously, these divisive methods work *top-down*. Example techniques for these classes are hierarchical agglomerative methods and hierarchical `k-means` (see [34]). A key contribution of this thesis is an empirical comparison of both kinds of methods.

Lastly, we can differentiate between *online* and *batch* approaches. Batch approaches use all available items at once to optimize the cluster assignment. On the contrary, online approaches use only small subsets of the available data to update parameters iteratively. This is necessary in cases when main memory is too small to store all the data, or in streaming data applications, where samples arrive one at a time and every sample is seen only once (see [4]). Another benefit of the online setting is the interleaved adaptation of different internal system parameters.

2.1.1 *Notation*

Assume a dataset $D = \{x_1, \ldots, x_N\}$ comprising N data items, here feature vectors of dimension d, and a partitions of these into a set of Clusters, $K = \{k_i \subseteq D \mid i = 1, \ldots, m\}$. Let n_i be the number of samples of the i-th cluster, $v_i = \frac{1}{n_i} \cdot \sum_{j=1}^{n_i} x_j^i$ the centroid of the i-th cluster, where x_j^i is the j-th data point belonging to the i-th cluster. Lastly, let $d(x, y)$ be the distance between two data points. Unless stated otherwise we assume euclidean distance (similar to [46]).

Hierarchical clustering is usually represented by a tree, representing the nested series of partitions. The used notation is derived from [13]. Let S denote the set of all tree nodes. Each node of the tree $s \in S$ is associated with a set of data points $k_s \subseteq D$. The number of elements in the cluster k_s is represented by $n_s = |k_s|$. The nodes' children are denoted by $c(s) \subset S$ and partition all the data points of the parent node such that $k_s = \cup_{r \in c(s)} k_r$. The leaf nodes of the tree correspond to the data points of the dataset split into N singleton clusters.

2.1.2 `k-means` *Algorithm*

`k-means` is the simplest and most commonly used clustering algorithm. It minimizes a squared error criterion

$$\sum_{j=1}^{m} \sum_{i=1}^{n_j} d(x_i^j, v_j)^2 \,. \tag{2.1}$$

It starts with a random initial partitioning and keeps reassigning the data points to clusters, based on the similarity between the data point and the cluster centroid (see Algorithm 1). Based on this the cluster centroids are updated. This is repeated until a convergence criterion is met (e.g. no reassignment of any data point to a

different cluster, or the squared error ceases to decrease significantly over multiple iterations; see [34]). The use of the euclidean distance in the cluster assignment step (2.a) arises from the implicit assumption that the overall distribution of the data can be decomposed into a mixture of isotropic Gaussians (see Subsection 3.3.2 for a derivation).

Algorithm 1 `k-means`

1. Initialize cluster centers $v_i, i = 1, ...m$ with randomly-chosen datapoints $x \in D$.
2. For a prespecified number of iterations or until convergence:
   a) For $j = 1$ to m:

$$k_j = \{x \in D : d(x, v_j) \leqslant d(x, v_r) \text{ for all } r \neq j\}$$

   b) For $j = 1$ to m:

$$v_j = \text{centroid}(k_j) = \frac{1}{n_j} \cdot \sum_{i=1}^{n_j} x_i^j$$

The `k-means` algorithm is popular, because it is very easy to implement and fast (time complexity in $O(Nmd)$, which is linear in N). A major problem is its sensitivity to the selection of the initial partition. While the algorithm has been proven to converge to a local minimum of the squared error criterion function, this local minimum might be of low quality in a global sense (see [48]). In practice, the `k-means` algorithm is usually run multiple times and the solution with the lowest squared error is chosen (see Equation (2.1)). Also, `k-means` is not as versatile as hierarchical agglomerative algorithms, because the resulting clusters are always hyperspherical in shape (see [34]).

2.1.3 *Hierarchical Agglomerative Clustering*

Conventional hierarchical agglomerative clustering algorithms work by first computing a complete matrix of distances between all data points in D and then, starting from singleton clusters containing only one data point each, using this matrix to sequentially merge elements together (bottom-up). Let $k_i = \{x_i\}, i = 1, ..., N$ be the disjoint singleton clusters. The distance matrix $[D_{ij}]$, $i, j = 1, \ldots, m$, defines the pairwise distances between clusters k_i and k_j, which initially is the distance between the data points x_i and x_j. In each iteration the two clusters k_i and k_j with minimum distance form a new cluster k_l, and the distance from k_l to each of the remaining clusters is updated. This distance update is done with a specific linkage method,

which are described below the algorithm. Since the distance measure is assumed to be symmetric, only the upper triangular component of the proximity matrix needs to be stored (see [11]). To simplify notation d_{ij} refers to the unique entry given by $d_{min(i,j),max(i,j)}$.

Algorithm 2 `Hierarchical Agglomerative Clustering`

1. $S := \{k_1, k_2, \ldots, k_N\}$
2. For each $(k_i, k_j) \in S^2$ compute $d_{ij} := d(x_i, x_j)$
3. For $l = N + 1$ to $2N - 1$:
   a) $(i^*, j^*) = \underset{(i,j) \in S^2}{\operatorname{argmin}} d_{ij}$
   b) Set $k_l := k_{i^*} \cup k_{j^*}$ and $n_l := n_{i^*} + n_{j^*}$
   c) $S := S \setminus \{k_{i^*}, k_{j^*}\} \cup k_l$
   d) For each $k_h \in S \setminus k_l$:
   compute $d_{hl} := f(d_{hi^*}, d_{hj^*}, n_h, n_l, k_h, k_l)$

The general algorithm has now the form described in Algorithm 2 (see [11]). The specific type of hierarchical agglomerative clustering is defined by the choice of a linkage method and a corresponding function f.

- *Single linkage* uses the minimum distance of data samples between two clusters as the distance between those clusters.
  $f(d_{hi^*}, d_{hj^*}, n_h, n_l, k_h, k_l) := \min_{x \in k_h, y \in k_l} d(x, y) = \min(d_{hi^*}, d_{hj^*})$
- *Complete linkage* uses the maximum distance of data samples between two clusters as the distance between those clusters.
  $f(d_{hi^*}, d_{hj^*}, n_h, n_l, k_h, k_l) := \max_{x \in k_h, y \in k_l} d(x, y) = \max(d_{hi^*}, d_{hj^*})$
- *Average linkage* uses the average distance of data samples between two clusters as the distance between those clusters.
  $f(d_{hi^*}, d_{hj^*}, n_h, n_l, k_h, k_l) := \frac{1}{n_h \cdot n_l} \cdot \sum_{x \in k_h} \sum_{y \in k_l} d(x, y)$

A key problem with agglomerative clustering is scalability. Unfortunately, all conventional hierarchical agglomerative clustering algorithms have a complexity in $O(N^2 d)$, which for instance makes them inapplicable for applications like content-based image clustering of more then 10.000 images (see [42]).

2.2 CLUSTER EVALUATION

Usually there is a semantic grouping that would be the right one. Since clustering is mostly an unsupervised process, correspondence to this semantic grouping cannot be ensured. Therefore, the evaluation of the clustering results is even more important (see [36, 39]).

Cluster validity evaluation is very hard, because clustering is subjective in nature. The same data set may need to be partitioned differently for different purposes. Jain et al. give the example of an *elephant*, a *tuna fish*, and a *whale*. Whales and elephants form a cluster of *mammals*. However, the user may be interested in the concept *living in water*. Then whale and tuna fish should be clustered together. This subjectivity is typically incorporated by domain knowledge, say by choosing certain features, similarity measures or clustering algorithms.

Maulik and Bandyopadhyay state two fundamental questions that need to be addressed in clustering systems. First, how many clusters are actually present in the data, and second, how good is the clustering itself. For our purpose the question regarding the number of clusters is not important, because we will use a fixed number in our proposed hierarchical algorithm. However, the question of cluster quality remains and gives rise to cluster validity analysis. Jain et al. state three types of validation studies:

- *Internal* examination of validity tries to determine if the outcome of clustering is intrinsically appropriate for the data.
- *External* assessment of validity compares the clustering result to an *a priori* structure (ground-truth).
- A *relative* test compares two clustering results and measures their relative quality.

In this thesis we will rely on external criteria only, since they are easier to interpret and more likely to differentiate bad from good results by comparing the generated clustering against a target partitioning. Internal criteria are feature space dependent and measure how compact and how widely separated the clusters are. But for the task of image clustering compact and well separated clusters are only of value if they correspond to semantic classes (which is measured by external measures). Some of the most popular internal validity indices are *Dunn's Index* [21], *Davies Bouldin Index* [16] and the *Calinki-Harabasz-Index* [10] (see [36, 39, 42]).

External Validity Measures

In the following, some external validity measures are introduced, which will be used in the experiments in chapter 4 (see [13, 46]). The most commonly used external

measures are *Purity*, *Weighted Purity* and *Entropy*, which are defined below. Assume a partition of all data items into a set of semantic groups or classes, $C = \{c_i \subseteq D \mid i = 1, \ldots, n\}$. Also let A be the contingency matrix produced by the clustering algorithm, representing its solution, such that $A = [a_{ij}]$, $i = 1, \ldots, n$ and $j = 1, \ldots, m$, where a_{ij} is the number of data points that are members of both class c_i and of cluster k_j. Finally, let $n_j = \sum_{i=1}^{n} a_{ij}$ be the size of cluster k_j. Then,

$$\begin{aligned} \text{Purity} &= \frac{1}{m} \sum_{j=1}^{m} \frac{\max_{i=1,\ldots,n} a_{ij}}{n_j} , \\ \text{Weighted Purity} &= \sum_{j=1}^{m} \frac{n_j}{N} \frac{\max_{i=1,\ldots,n} a_{ij}}{n_j} , \\ \text{Entropy} &= \sum_{j=1}^{m} \frac{n_j}{N} \left(-\frac{1}{\log n} \sum_{i=1}^{n} \frac{a_{ij}}{n_j} \log \frac{a_{ij}}{n_j}\right) . \end{aligned} \tag{2.2}$$

Purity represents the ratio of the dominant semantic class in each cluster averaged over all clusters. For Weighted Purity we do not use the average purity over all clusters, but an average weighted by the cluster sizes. For (Weighted) Purity a larger value means that the cluster is a "purer" subset of the dominant semantic class. Since Entropy also considers the distribution of semantic classes in a cluster, it can be seen as a more comprehensive measure than Purity. Note that contrary to (Weighted) Purity, an entropy value near 0 means the cluster is comprised mainly of one single class, while a value near 1 implies that the cluster contains a uniform mixture of all classes (see [13]).

Another external measure is the *V-Measure* (proposed in [46]), which is based on conditional entropy. In Information Theory, the conditional entropy $H(Y|X)$ quantifies the remaining entropy (i.e uncertainty) of a random variable Y given that the value of another random variable X is known (see [15]). The V-Measure explicitly measures how successful the two complementary criteria of *homogeneity* and *completeness* have been satisfied. A clustering result satisfies homogeneity if all of its clusters contain only data points which are members of a single class. It satisfies completeness if all the data points that are members of a given class are elements of the same cluster. Note that these criteria are complementary by considering two degenerate clustering solution. First, assigning all data points to one single large cluster gives perfect completeness, but is as inhomogeneous as possible. Second, assigning each data point to a distinct cluster satisfies perfect homogeneity, but is poor in terms of completeness.

Formally, this is expressed in terms of conditional entropy with some normalization effort to satisfy the convention of 1 being desirable and 0 undesirable.

homogeneity

$$H(C|K) = -\sum_{k=1}^{|K|}\sum_{c=1}^{|C|}\frac{a_{ck}}{N}\log\frac{a_{ck}}{\sum_{c=1}^{|C|}a_{ck}} \qquad H(C) = -\sum_{c=1}^{|C|}\frac{\sum_{k=1}^{|K|}a_{ck}}{N}\log\frac{\sum_{k=1}^{|K|}a_{ck}}{N}$$

$$h = \begin{cases} 1 & \text{if } H(C,K) = 0\,, \\ 1 - \frac{H(C|K)}{H(C)} & \text{else} \end{cases}$$

completeness

$$H(K|C) = -\sum_{c=1}^{|C|}\sum_{k=1}^{|K|}\frac{a_{ck}}{N}\log\frac{a_{ck}}{\sum_{k=1}^{|K|}a_{ck}} \qquad H(K) = -\sum_{k=1}^{|K|}\frac{\sum_{c=1}^{|C|}a_{ck}}{N}\log\frac{\sum_{c=1}^{|C|}a_{ck}}{N}$$

$$c = \begin{cases} 1 & \text{if } H(K,C) = 0\,, \\ 1 - \frac{H(K|C)}{H(K)} & \text{else.} \end{cases}$$

Finally, the V-Measure is computed as the harmonic mean of distinct homogeneity and completeness scores, allowing them to be weighted to favor one's contribution over the other (similar to the popular F-Measure for precision and recall):

$$V_\beta = \frac{(1+\beta)\cdot h\cdot c}{(\beta\cdot h)+c}\,.$$

We observe, that $H(C|K)$ is directly related to what we referred to as Entropy above (see Equation (2.2)), because $\sum_{c=1}^{|C|}a_{ck} = n_k$. Also $\frac{a_{ck}}{N} = P(\text{cluster } k, \text{class } c) = P(\text{cluster } k)\cdot P(\text{class } c \mid \text{cluster } k) = \frac{n_k}{N}\cdot\frac{a_{ij}}{n_j}$ again as in the Entropy definition by Rosenberg and Hirschberg (see [46]). Therefore, the only difference is the normalization factor of $1/\log n$ in the Entropy definition.

2.3 BALANCED CLUSTERING

In many applications having empty or very small clusters is undesirable and it is required that clusters obtained are balanced, meaning of approximately the same size. This balancing requirement comes from the associated application rather than from the inherent properties of the data, and helps in making the clusters actually useful and actionable. Some specific examples given by Banerjee and Ghosh are Direct Marketing, Category Management, Clustering of Documents and Balanced Clustering in Energy Aware Sensor Networks. In content-based image clustering, which is the focus of this thesis, balancing helps in making the clusters useful for browsing because the user orientation in the dataset largely relies on the regular structure of the clustering tree.

Clustering methods are said to be *dilating* if individual elements not yet in groups are more likely to form new groups instead of being merged to existing ones. Dilating methods like complete linkage have the advantage that they produce balanced trees,

which are useful for browsing. In contrast, single linkage is known to create very deep and unbalanced trees (see [8, 13]).

Balancing constrained clustering may yield solutions that are of poorer quality than non-constrained approaches, when measured by a data-driven criterion such as the squared error objective function (`k-means`), even though these same solutions are more preferable from the application viewpoint. Surprisingly, balanced clustering has been proven empirically to produce comparable and sometimes even better results than their non-constrained counterparts despite the assumption of all clusters having the same prior probability/number of samples. This has been explained by the fact that balancing provides a form of regularization that seems to avoid low-quality local minima stemming from poor initialization (see [5]). In addition to application requirements, balancing can be helpful because it tends to decrease sensitivity to initialization and avoid outlier clusters (highly underutilized representatives) from forming, and thus has a beneficial regularizing effect even in situations where balancing is not a requirement. The initialization dependence gets worse when input dimensionality as well as the number of clusters is high, thereby vastly expanding the solution space (see [4]). It has been shown that even tough `k-means` has an implicit way of preventing highly skewed clusters (see [37]), it often generates empty or extremely small clusters, especially in high-dimensional space (see [3]).

Banerjee and Ghosh summarize existing approaches to balanced clustering in [3].

General Approaches

One approach is to convert the clustering problem into a graph partitioning problem, which is then solved by a min-cut algorithm. However, this method does not scale well, due to its $O(N^2)$ complexity.

Banerjee and Ghosh present an $O(kN \log N)$ scheme, that can be broken down into three steps - sampling, soft-balanced clustering of the sampled set and populating, and refining the clusters while satisfying balancing constraints. The algorithm for populating the clusters is based on a generalization of the "stable marriage" problem, whereas the refinement algorithm is a constrained iterative relocation scheme (see [3, 4, 5]).

Hierarchical Agglomerative Approaches

Agglomerative clustering methods can be adapted so that once a cluster reaches a certain size in the bottom-up agglomeration process, it can be removed from further consideration.

A similar approach was proposed by Borth et al., who introduce a weighted penalty term depending on the cluster size (see [8]). Thereby, they enhance Average Linkage to the so-called *Balanced linkage*. Formally, we obtain $f(d_{hi^*}, d_{hj^*}, n_h, n_l, k_h, k_l) := \frac{1}{n_h \cdot n_l} \cdot \sum_{x \in k_h} \sum_{y \in k_l} d(x, y) + \alpha \cdot (n_h + n_l)$ (see Algorithm 2).

A major problem of hierarchical agglomerative approaches is scalability, since they require a distance matrix of size $O(N^2)$.

`k-means`-based Approaches

A first approach is to iterate over `k-means`, but to do the cluster assignment (step 2.a in Algorithm 1) by solving a minimum cost flow problem satisfying balancing constraints. This approach is $O(N^3)$ and thus has even poorer scaling properties than the hierarchical agglomerative methods above (see [3]).

Balancing can also be achieved by formulating the cluster assignment step of `k-means` as a Linear Program and explicitly adding constraints to the optimization problem, requiring that each cluster contains at least a certain number of points (see [9]).

A last class of algorithms is called *Frequency Sensitive Competitive Learning* algorithms (FSCL). FSCL was originally formulated to remedy the problem of underutilization of parts of a codebook in vector quantization (see [2]). The employed distance measure is weighted by the number of assignment to the respective cluster. Formally, a data point x is assigned to the k^*-cluster, such that $k^* = \operatorname{argmin}_i[F(n_i) \cdot d(v_i, x)]$, where n_i is the cluster size, v_i the cluster centroid of the i-th cluster and F an increasing function of the cluster size. During the next iterations the distance measure for large clusters increases, hereby equalizing the number of assignments over time (see [47]). Although frequency sensitive assignments can give fairly balanced clusters in practice, there is no obvious way to guarantee that every cluster will have at least a certain number of points (see [5]). A convergence study on FSCL algorithms has shown that only conditions on the learning rate must be imposed to guarantee convergence to a locally optimal solution (see [25]).

3 CLUSTERING FOR IMAGE BROWSING

While the previous chapter dealt with clustering and balancing strategies in general, we turn towards the specific application of content-based image clustering and browsing. This chapter first discusses related work on this topic. Then, particular requirements of clustering posed by the application of image browsing are summarized. Third, an approach for efficient and scalable image clustering is proposed based on a balanced online variant of the hierarchical `k-means` algorithm. We call this method *hierarchical balanced online k-means*, or short `hbo-k-means`. It constitutes one of the main contributions of this thesis.

3.1 CONTENT-BASED IMAGE CLUSTERING

This section gives a brief overview of related work on content-based image clustering. Due to space limitations, we only review work most related to ours.

Navidgator, a system for similarity-based browsing of multimedia databases proposed by Borth et al., provides the basis for this thesis (see [8]). A *balanced linkage* method for hierarchical agglomerative clustering is introduced to enforce balanced tree structures. In addition, a Graphical User Interface is presented to navigate through a database of video keyframes, offering a coarse-to-fine similarity-based view of the grouped content. The authors also observed that a similarity-based one-dimensional sequence is obtained as a by-product in the proposed clustering process. Parts of this sequence correspond to clusters of high similarity at lower levels of the tree. The concept of these parts containing similar content is called *Content-Stripes*. This concept is illustrated in Figure 3. Content-Stripes replace the need to additionally compute spatial arrangements for images within a cluster like it is done with similarity pyramids (see [11]). Therefore, a similarity-based order is created in addition to the hierarchical clustering structure in one single step instead of separating these tasks (see [8]). In Section 3.3 a `k-means`-based clustering approach will be proposed to enhance the Navidgator back-end in terms of balancing and scalability.

Chen et al. introduce the concept of a *similarity pyramid* for content-based browsing of large image databases (see [11]). The similarity pyramid represents the database whereas different layers in the pyramid correspond to varying levels of detail, such that similar images are located nearby each other on a 2D grid of images. They use hierarchical agglomerative clustering with a sparse proximity matrix, only containing the closest M matches of each sample ($M < N$) to reduce the complexity to $O(MN)$. This approach is called *fast sparse clustering*. However, to find those M matches an

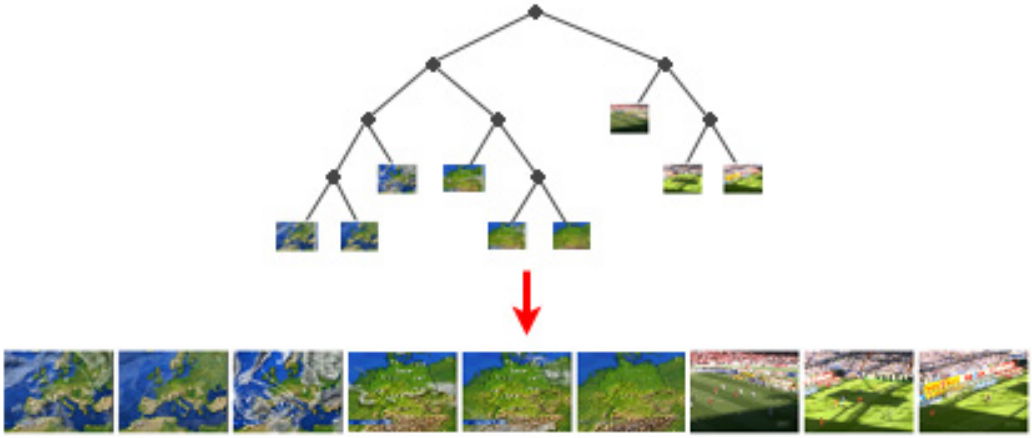


Figure 3: A tree representation of a sample clustering, transformed to a Content-Stripe displaying similarity clusters in an one-dimensional order (image and caption from [8]).

additional top-down clustering of the whole data set is required. Also, the clustering algorithm only generates binary tree structures,which need to be transformed into *quad-trees* (with a branch factor of 4). Here, balancing is incorporated in the cost-function, which optimizes the mapping of the quad-tree to the pyramid, minimizing spatial variations within a cluster.

Chen et al. try to bridge the semantic gap by clustering, stating the hypothesis that "semantically similar images tend to be clustered in some feature space" (see [13]). They introduce *CLUster-based rEtrieval of images by unsupervised learning* (CLUE), which attempts to capture semantic concepts by retrieving image clusters instead of an ordered list of images as conventional image-retrieval systems do. Käster et al. add the applicability of image clustering techniques to scale up content-based image retrieval systems (see [36]). Clustering also can be used to restrict the search space and to avoid exhaustive comparisons. This does not only speed up the search process, but may also improve search results.

Goldberger et al. propose continuous image modeling based on mixtures of Gaussian densities, grouping pixels into coherent regions in feature space (see [27, 28]). These are then clustered bottom-up, using the *Information Bottleneck principle*: Images are grouped so that the mutual information between the clusters and the image content is maximally preserved. Due to a Monte Carlo simulation for the Kullback-Leibler-distance approximation the method is computationally expensive and therefore not applicable to large-scale image clustering.

Yeung et al. use a *time-constrained clustering* to extract story units from videos for browsing and navigation (see [56]). Better video organization is achieved by clustering similar video shots into scenes, using both visual similarities and temporal localities of the shots.

Finally, Nister and Stewenius propose a large-scale object recognition scheme which builds on local region descriptors (see [45]) These are hierarchically quantized in a codebook or *vocabulary tree*. This vocabulary tree allows are larger and more discriminatory vocabulary to be used efficiently.

3.2 PARTICULAR REQUIREMENTS

Basically, there are three particular requirements that should be met by clustering algorithms for image browsing, namely quality, scalability and balancing.

Quality

First, the produced clusters should be of be a certain *quality* to be actually useful for the task. That is, the partitioning into clusters must adhere to semantic classes which are helpful to the user. These semantic classes could consist of images of the same object (e.g. "spider"), or of the same picture series, or showing the same semantic concept like "france" or "dog". Recall that the partitioning into the "correct" clusters is non-trivial due to the subjectivity of the task and the semantic gap (see Chapter 1).

Scalability

Second, the clustering algorithm needs to be able to cluster large-scale image collections, using only a reasonable amount of resources. Hence, *scalability* is required, which is mainly imposed by the advent of large image databases not only in commercial but also in private environments. Whereas the systems proposed in [8] and [11] have only been tested on datasets with up to 10.000 images, new clustering approaches should be able to process up to 1 million images. Therefore, all approaches in $\Omega(N^2)$ (particularly, hierarchical agglomerative clustering) are not applicable.

Balancing

Third, we demand that the clusters obtained are of approximately the same size, which we refer to as *balancing*. As already motivated in Section 2.3, this requirement arises from application needs rather than from inherent properties of the data, i.e. balancing helps in making the clusters useful for browsing. Balanced trees are more likely to represent the semantic complexity of the dataset in a comprehensible way than deep and unbalanced ones, because they try to equally distribute the semantic complexity on each single level. In addition, regular structures help the user browse an image collection to orient themselves in the dataset. Unbalanced trees contain leaves on very different levels. Thus, very small and specific and very large and unspecific

clusters may occur on the same level. Therefore, the depth in the tree cannot be utilized properly for user orientation (which is done for instance in Navidgator). Consequently, the individual interaction possibilities have different effects depending on the current node of the tree. This confuses the user while browsing and leads to a bad user experience. We assume that it is intuitive to split the clusters into equal parts on each level to appropriately distribute the inherent complexity of the data. Thus, unbalanced trees can be said to be result of non-intuitive splits on upper levels of the tree. In conclusion, unbalancing takes away the regular structure of the tree on which the user orientation implicitly relies.

3.3 OUR PROPOSED APPROACH: `hbo-k-means`

Before going into detail how balancing can be enforced in a simple and efficient way, we take a look at the overall top-down structure of the hierarchical `k-means` algorithm which will be applied to clustering of image collections (see [45]).

3.3.1 *General Functionality*

Let us first look at the general approach to produce a nested series of partitions, which will be represented by a tree structure. Note that not only the pure cluster partitioning is of interest but also the relationship between different clustering levels (subcluster partitions etc.; see [8]).

An algorithmic description of the *hierarchical k-means* algorithm (`h-k-means`) can be found in Algorithm 3.

Algorithm 3 Hierarchical `k-means` (`h-k-means`)

1. Initialize a tree structure with a root node containing the whole dataset D.
2. While the tree has leaf nodes that contain more than *threshold* images do:
   a) Choose one of these nodes and use the `k-means` algorithm to partition its images (using a prespecified *branch factor* as the number of clusters).
   b) Use these partitions to create children of the current node.

In the experiments, a *threshold* of 4 is used, meaning that a leaf node is not subdivided further if comprising four or less images.

3.3.2 *Efficient Balancing*

In the k-means algorithm a distance measure (namely the euclidean distance) is used to decide which cluster centroid v_i is nearest to a given data point x (step 2.a of Algorithm 1). Assume that x is distributed according to a mixture of isotropic Gaussians with a uniform prior. Thus, covariance matrices $\Sigma_i = I$ are used, where I is the identity matrix. Then, the data point is assigned to cluster k^*-th cluster, such that:

$$\begin{aligned}
k^* &= \operatorname{argmax}_i \ \log P(k_i|x) \\
&= \operatorname{argmax}_i \ \log \frac{P(x|k_i) \cdot P(k_i)}{P(x)} \\
&= \operatorname{argmax}_i \ \log P(x|k_i) \\
&= \operatorname{argmax}_i \ \log[\frac{1}{\sqrt{(2\pi)^d |I|}} \exp(-\frac{1}{2}(x - v_i)^T I^{-1}(x - v_i))] \\
&= \operatorname{argmax}_i \ -\frac{1}{2} d(x, v_i)^2 + \text{const.} \\
&= \operatorname{argmax}_i \ -d(x, v_i) \\
&= \operatorname{argmin}_i \ d(x, v_i) \ .
\end{aligned}$$

Frequency Sensitive Competitive Learning algorithms (FSCL) utilize the use of a distance measure to incorporate balancing. This class of algorithms multiplies the distance term by a fairness function F, which is an increasing function of the cluster size. Formally, x is assigned to the k^*-th cluster, such that

$$k^* = \operatorname{argmin}_i [F(n_i) \cdot d(x, v_i)] \ ,$$

where n_i is the cluster size and v_i the cluster centroid of the i-th cluster. Usually F is the identity function (see [25, 47]). A similar approach was proposed by Banerjee and Ghosh, where a frequency sensitive learning mechanism was derived from a mixture of Gaussians framework by making each of the Gaussians *shrink* in proportion to the number of points that have been assigned to it [4].

Here *shrinking* means that the density gets more peaked by using covariance matrices $\Sigma_i = \frac{1}{n_i} I$, where again I is the identity matrix, and maximizing the log-

likelihood of a particular point x with respect to this Gaussian, results in assigning point x to cluster k^* such that:

$$\begin{aligned}
k^* &= \text{argmax}_i \ \log P(k_i|x) \\
&= \text{argmax}_i \ \log \frac{P(x|k_i) \cdot P(k_i)}{P(x)} \\
&= \text{argmax}_i \ \log P(x|k_i) \\
&= \text{argmax}_i \ \log[\frac{1}{\sqrt{(2\pi)^d |\Sigma_i|}} \exp(-\frac{1}{2}(x-\nu_i)^T \Sigma_i^{-1}(x-\nu_i))] \\
&= \text{argmax}_i \ -\frac{n_i}{2} d(x,\nu_i)^2 + \frac{d}{2}\log n_i - \frac{d}{2}\log 2\pi \\
&= \text{argmin}_i \ n_i \cdot d(x,\nu_i)^2 - d \log n_i \ ,
\end{aligned}$$

where ν_i is the centroid of the i-th cluster and d is the dimensionality of the data. Thus, formal treatment results in an additional term, namely $-d \log n_i$ and it has been shown that FSCL methods essentially perform frequency sensitive cluster deformation by shrinking Gaussians (see [4]).

While these balancing approaches were demonstrated to work well on low-dimensional synthetic datasets, they were experienced as very unstable in high-dimensional feature space. Especially applied in batch-based `k-means`, these approaches were experienced to deteriorate cluster balancing due to a very high sensitivity to very small and very large clusters. For instance, if the clustering was initialized with regular `k-means`, sometimes very small outlier clusters occur (using a random initialization this is usually worse). The frequency sensitive distances to these clusters are so small that in one iteration (almost) all samples are assigned to them, creating a number of other empty or very small clusters, which again attract all samples in the next iteration, leading to degenerate solutions of extremely inbalanced clusters. See table 1 for an illustrating sample run. To some extent, this can be blamed on what we call "temporal bias", because in each iteration all N data points are assigned to clusters, resulting in very different clusters and cluster sizes compared to the previous iteration. Using an online approach it is possible to reestimate cluster sizes after each new assignment, yielding a more stable behavior. Unfortunately, "oscillations" of very small clusters becoming very large and vice versa still occur even though less often. Additionally, these methods seem to be very dependent on the initialization and tuning parameters where required. Using the same tuning parameters produced very different results on different datasets ranging from satisfying to degenerate solutions. Therefore, a different approach to incorporate balancing is proposed in the following.

Strategy 1: Online Approach

The concept of "temporal bias" was introduced above to explain the instability of batch-based approaches. To prevent oscillation of cluster sizes, cluster assignments are

ITER.	CLUSTER SIZES												
1	100	100	100	100	100	100	100	100	100	100	100	100	100
2	37	59	16	358	68	228	154	141	84	23	26	37	69
3	111	22	47	2	55	1	66	120	11	324	336	76	129
4	347	1	3	2	3	1	36	869	0	0	0	38	0
5	0	103	267	71	61	1	615	0	0	0	0	182	0
6	1300	0	0	0	0	0	0	0	0	0	0	0	0

Table 1: A sample run illustrating the "oscillations" in cluster sizes and convergence to a degenerate solution. This can be partly blamed on what we call "temporal bias".

interleaved with cluster size estimation. Essentially, this results in an online approach that reestimates cluster sizes after each new assignment. Our proposed method uses this online approach and will be demonstrated to be more stable than the FSCL methods introduced above (see Chapter 4).

Strategy 2: Frequency Sensitive Prior Adaption

Instead of cluster deformation as in the derivation above, the key idea of this new approach is to adapt the prior probability of the cluster $P(k_i)$ to favor small clusters over larger ones. Whereas Banerjee and Ghosh assumed a uniform prior (which vanishes because it is independent of i; see [4]), $P(k_i)$ is now assumed to decrease for growing clusters, formally $P(k_i) = \exp(-n_i)/\sum_{j=1}^{m} \exp(-n_j)$ while the *shape* of clusters remains unchanged, i.e. $P(x|k_i) = N(x, \nu_i, I)$. Now we obtain:

$$
\begin{aligned}
k^* &= \operatorname{argmax}_i \ \log P(k_i|x) \\
&= \operatorname{argmax}_i \ \log \frac{P(x|k_i) \cdot P(k_i)}{P(x)} \\
&= \operatorname{argmax}_i \ \log P(k_i) + \log P(x|k_i) \\
&= \operatorname{argmax}_i \ \log[\exp(-n_i)/\sum_{j=1}^{m} \exp(-n_j)] + \log N(x, \nu_i, I) \\
&= \operatorname{argmax}_i \ -n_i - d(x, \nu_i)^2 + \text{const.} \\
&= \operatorname{argmin}_i \ n_i + d(x, \nu_i)^2 \ .
\end{aligned}
$$

We see that a change of prior leads to an *additive bias* of n_i instead of a multiplicative one.

This single additive bias serves as the basis for our proposed approach. To have control over the balancing behavior, a variable α is introduced to weight the influence of the cluster size. This approach has already been suggested by Borth et al., who - instead of multiplying the distances by cluster size - introduced an additive weighted penalty term depending on the cluster size (originally to enhance average linkage hierarchical agglomerative clustering). Formally, $k^* = \text{argmin}_i[d(v_i, x) + \alpha \cdot n_i]$ for $0 \leqslant \alpha < \infty$. Although this weight needs to be tuned to achieve the expected behavior, this method will be demonstrated to be much more stable for both batch and online approaches in our experiments compared to the FSCL methods discussed above. In the original paper the α value was empirically evaluated and set to $\alpha = 0.01$.

However, this fixed value of α is feature- and data-dependent and certainly not intuitive. Thus, the approach is extended to the following more flexible model, which allows to continuously trade off true feature similarity against balancing constraints:

$$k^* = \underset{i}{\text{argmin}}[(1-\beta) \cdot d(v_i, x) + \beta \cdot \alpha \cdot n_i] \ ,$$

where a reasonable value for α is estimated from the given data and $0 \leqslant \beta \leqslant 1$ is the only true parameter. For a value of $\beta = 0.5$ both distance-based similarity and cluster-size-based penalty should be weighted equally "on average", which means that for the average cluster size of N/m and $x \in D$ we postulate that:

$$\text{average}[d(v_i, x)] = \alpha \cdot \text{average}[n_i] = \alpha \cdot \frac{N}{m} \ .$$

average$[d(v_i, x)]$ is estimated by computing and averaging a sufficiently large number of distances between random data points in the given feature space, referred to as `avg_randomdist`. Thus, it is possible compute $\alpha = \texttt{avg\_randomdist} \cdot \frac{m}{N}$. Now, the convenient parameter β allows to continually and intuitively trade off true feature similarity against balancing constraints.

Balanced Online `k-means`

Algorithm 4 is a description of our *balanced online `k-means`* algorithm (`bo-k-means`), which will provide the basis for our proposed `hbo-k-means` algorithm for clustering image collections. Like the other introduced approaches, `bo-k-means` belongs to the class of Frequency Sensitive Competitive Learning algorithms. However, an additive bias is used instead of a multiplicative one as described above.

The *learning rate* is an important detail in online learning algorithms. In Step 5.a)v it was defined by $v_{k^*} := v_{k^*} + (1/n_{k^*}) \cdot (x - v_{k^*})$, which is just a recursive form to compute the centroid of n_{k^*} data points. If the data is obtained from a stationary process (i.e. no change in parameters of the underlying generative model over time), then the centroid (as computed by the above recursion) will converge, and does not need updating after a sufficiently large number of iterations (see [4]).

Algorithm 4 Balanced Online k-means (bo-k-means)

1. Initialize cluster centers $v_i, i = 1, ...m$ with randomly-chosen data points $x \in D$.
2. Compute the cluster sizes $n_i, i = 1, \ldots, m$ by assigning each data point using euclidean distance.
3. Estimate avg_randomdist by averaging a sufficiently large number of distances between random data points and compute $\alpha = \texttt{avg\_randomdist} \cdot \frac{m}{N}$.
4. Precompute a random permutation of the dataset, which defines the order of the data points in the loop. Note that it might be necessary to iterate over the dataset multiple times.
5. For a prespecified number of iterations or until convergence:
   a) For all data points in the order of the random permutation:
      i. Let x be the current data point and l the index of the cluster it is currently assigned to.
      ii. Compute to which cluster x should be assigned to:
      $$k^* = \underset{i}{\operatorname{argmin}}[(1-\beta) \cdot d(v_i, x) + \beta \cdot \alpha \cdot n_i] \, .$$
      iii. Update cluster sizes:
      $$n_{k^*} := n_{k^*} + 1 \, , \qquad n_l := n_l - 1 \, .$$
      iv. Update the cluster assignment of x to cluster k^*.
      v. Update the centroid of cluster k^*:
      $$v_{k^*} := v_{k^*} + (1/n_{k^*}) \cdot (x - v_{k^*}) \, .$$
   b) After each epoch (full run over the dataset):
      i. Check for empty clusters and reinitialize the cluster centroid randomly if necessary.
      ii. Check for convergence by computing the ratio of different assignments compared to the previous epoch. If this ratio is below convergence-threshold return the cluster assignment.

We now call the variant of the `h-k-means` algorithm (see Algorithm 3), where the regular `k-means` in step 2.a is replaced with the just described `bo-k-means` algorithm `hbo-k-means`. It will become evident from the experiments in Chapter 4 that the proposed method is able to cluster large-scale image collections efficiently, producing extremely balanced clustering trees of high quality.

4

EXPERIMENTS

This chapter provides a description of the conducted experiments and their experimental results after describing the used datasets and feature descriptors.

In Experiment 1 `k-means` is compared to a number of hierarchical agglomerative approaches using multiple different features. Here, `k-means` will be demonstrated to be suitable for content-based image clustering.

The impact of a hierarchical top-down approach is quantitatively investigated in Experiment 2, where cluster quality is evaluated on different levels of the hierarchy. It will be shown that hierarchical `k-means` methods do not necessarily perform worse than hierarchical agglomerative approaches but yield substantial speed benefits.

The third experiment was conducted to demonstrate the impact of our proposed balancing approach `bo-k-means`. We will see that we can obtain balanced clusters efficiently while preserving or even improving cluster quality. In addition, the proposed balancing approach will be found to outperform FSCL methods in terms of stability.

In Experiment 4 the behavior of balancing approaches is examined in a hierarchical setting where our both our proposed `hbo-k-means` algorithm and a balanced hierarchical agglomerative clustering method are compared to their unbalanced counterparts.

Finally, the proposed method `hbo-k-means` is demonstrated to be scalable on a large-scale dataset and it will be shown that both cluster quality and balancing seem to be independent of the used branch factor.

4.1 DATASETS

Multiple clustering approaches are evaluated on six different datasets with different characteristics. *corel-13* and *corel-45* are subsets of different sizes of the well known Corel Photo CDs, which have become a de-facto standard in the field of content-based image retrieval (see [43]). They contain images from different countries and cities all over the world. The *netclean* dataset contains series of pornographic images (which also constitute the target clustering) and was provided by Netclean Technologies, a partner in the FIVES Project with the goal to ban child pornography from the internet (see [30]). Another dataset used is the *caltech-256* Object Category Dataset, which is intended to facilitate computer vision research (see [29]). It contains images of 256 different object categories. This dataset is of special importance, because first, hierarchical agglomerative approaches meet their computational limits with 30k samples and second, its clusters are not of equal size, which violates the implicit

assumption of all balancing approaches. From Chapter 1 it is known that content-based image clustering is also very important in the video domain. Thus, two more datasets are added. Both the *youtube* and the *youtube-large* dataset consist of randomly sampled keyframes of YouTube videos of different semantic categories or concepts which were inferred from user-generated tags. Table 2 summarizes the number of sample images, the number of clusters and the variation in cluster sizes for each dataset.

For all these datasets target categories are defined, which is used by external cluster validity measures. Note that these category labels are subjective in nature and partially represent very high-level concepts like "france" or "ireland". Identifying these concepts from raw images might even be very hard for humans in some cases. Sample images illustrating the intra-cluster variance are shown in Figures 4, 5, 6, 7, 8 and 9. Also, weaknesses of tagging approaches mentioned in chapter 1 can be observed. For example, one image of Figure 9 shows the American football star Michael Vick, who was convicted of dog fighting. Even images of cats or basketball show up, because these scenes were part of videos which were tagged with "dog" by a YouTube user. Due to the random keyframe extraction also black keyframes appear in the dataset. For obvious reasons is not possible to meaningfully assign them to a specific semantic cluster. Therefore, perfect results are not to be expected.

Figure 4: Sample images of the "singapor" category of the corel-13 dataset.

4.2 FEATURES

Several features are used in the experiments to largely cover the current state-of-the-art features available for content-based image systems. These features are described below.

Figure 5: Sample images of the "france" category of the corel-45 dataset.

Figure 6: Sample images of the 50-th category of the netclean dataset.

Figure 7: Sample images of the "spider" category of the caltech-256 dataset.

Figure 8: Sample images of the "soccer" category of the youtube dataset.

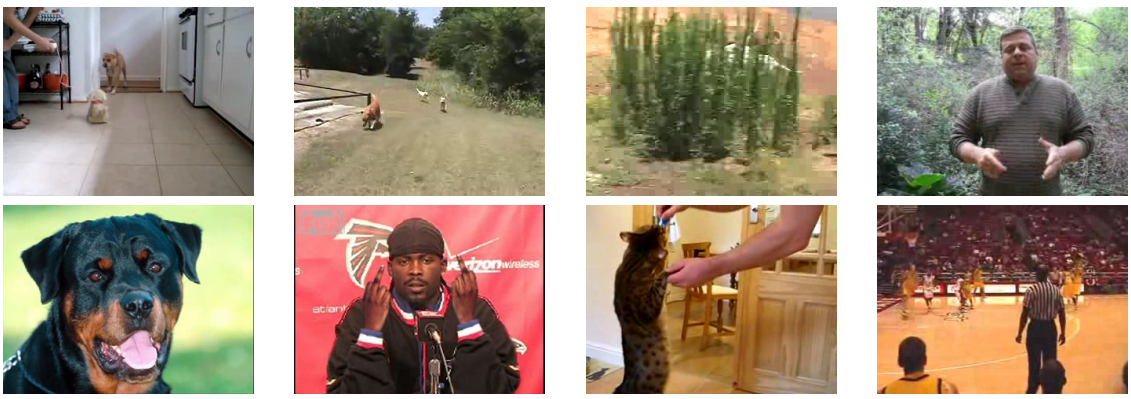


Figure 9: Sample images of the "dogs" category of the youtube-large dataset.

DATASET	#IMAGES / #CLUSTERS	CLUSTER SIZES	GROUND TRUTH PARTITIONING	EXAMPLE CATEGORIES
corel-13	1300 / 13	100	same country or city	ny-city, singapor, thailand
corel-45	4500 / 45	100	same country or city	japan, washington, kyoto
netclean	2000 / 200	10	same picture series, usually same location and person	serially numbered
caltech-256	29780 / 256	80-800	same object category	american-flag, boxing glove, fried-egg
youtube	5000 / 10	100	tagged with the same YouTube Tag	cats, helicopter, soccer
youtube-large	716266 / 225	226-8606	tagged with the same YouTube Tag	furniture, driver, tony-blair

Table 2: An Overview of all datasets used in this thesis.

4.2.1 *Color Histograms*

The *color histogram*, introduced by Swain and Ballard, is probably the most commonly used descriptor for images and describes the global distribution of colors in a picture. Statistically, a color histogram is a way to approximate the joint probability of the values of the three color channels (RGB, HSV, etc.). It is defined by:

$$h_{A,B,C}(a,b,c) = N \cdot P(A = a, B = b, C = c) \ ,$$

where N is the number of pixels in the image. Depending on the requirements, the individual axes of the color space can be quantized differently, determining the length of the descriptor. For convenience this 3D representation is usually transformed into a 1D vector (see [53]).

4.2.2 *Color Layout Descriptors*

One color descriptor of the Multimedia Content Description Interface (MPEG-7 Standard; see [49]) is the *color layout descriptor* (CLD). CLD is a compact 12-dimensional descriptor that uses representative colors on an 8x8 grid followed by a Discrete cosine transform (DCT) and encoding of the resulting coefficients. The feature extraction process consists of two parts; grid based representative color selection and DCT transform with quantization. An input picture is divided into 64 blocks and their average colors are computed. This partitioning process is important to guarantee resolution or scale invariance. The derived average colors are transformed into a series of coefficients by performing DCT. A few low-frequency coefficients are selected using zig-zag scanning and quantized to form a CLD (see [41]).

4.2.3 *Color Moments*

Stricker and Orengo state that it is hard to find an optimal color space quantization and that furthermore even an optimal quantization will produce unwanted quantization effects. Therefore they proposed instead of storing complete color distributions to store only their dominant features, and showed that this feature has the potential to outperform histogram-based methods in the robustness of the results as well as in terms of retrieval speed.

From probability theory we know that a probability distribution is uniquely characterized by its moments, resp. central moments. Thus, if we interpret the color distribution of an image as a probability distribution, the color distribution can be characterized by its moments as well. Stricker and Orengo propose to store the first three moments (average, variance and skewness) of each color channel. To make them comparable in terms of unit, the standard deviation and the third root of the skewness are used. If the value of the i-th color channel at the j-th image pixel is p_{ij}, then the average E_i, standard deviation σ_i and third root of the skewness s_i are (see [52]):

$$E_i = \frac{1}{N}\sum_{j=1}^{N} p_{ij}\,, \quad \sigma_i = \left(\frac{1}{N}\sum_{j=1}^{N}(p_{ij}-E_i)^2\right)^{\frac{1}{2}} \quad \text{and} \quad s_i = \left(\frac{1}{N}\sum_{j=1}^{N}(p_{ij}-E_i)^3\right)^{\frac{1}{3}},$$

which are concatenated to a 9-dimensional feature (HSV color space).

4.2.4 *Color Correlograms*

Whereas color histograms are fast to compute, they fail to describe the spatial distribution of colors the image. Several approaches were proposed to integrate spatial information with color histograms. One frequently used spatial correlation descriptor is the *correlogram* by Huang et al. (see [30, 31]).

The correlogram describes the spatial color correlation of pixels as a function of their spatial distance: Given a color c_i compute the probability $P(c_i, c_j, d_n)$ of finding a color c_j at distance d_n, using D8 distance also known as chessboard distance.

The *autocorrelogram* is a simplified version of the correlogram and describes only the probability of a pixel with color c_i to find colors from the same bin at distance d_n (see [32]). Both the idea of the autocorrelogram and the D8 distance are visualized in Figure 10.

Feng shows that correlation-based descriptors are more stable to color changes than color histograms (see [24]). Changes in color tend to have big effect on color histogram but noticeably less effect on correlation descriptors. In addition, he points out that changes in appearance such as cropping have less influence on correlation descriptors than on color histograms, and that correlation-based descriptors are more robust to changes in contrast and brightness (see [24, 30]).

Figure 10: The functionality of the autocorrelogram (image and caption from [30]).

The Modified Autocorrelogram

To overcome the complexity issues of correlograms and autocorrelograms, Hofmann and Ali propose a *modified autocorrelogram*. Both the modified autocorrelogram and the ordinary autocorrelogram describe the correlation of colors in an image. For autocorrelograms the probability is computed that pixels of a given color c_i have pixels of the same color at various distances. To formally define the autocorrelogram

let p_1 be the center pixel of the mask, p_2 any other pixel inside the mask and let k denote the distance (using the L_∞-norm) between p_1 and p_2. For a pixel $p = (x, y) \in I$, let $I(p)$ denote its color and let $I_c := \{p \mid I(p) = c\}$ (see [30, 32]):

$$\gamma_{c_i}^{(k)}(I) := P[p_2 \in I_{c_i} \mid p_1 \in I_{c_i}, \|p_1 - p_2\| = k] \,.$$

The modified autocorrelogram now neglects the distinction between various distances and computes the overall probability for the center pixel of the mask of having surrounding pixels in the same mask with the same color, or formally:

$$\gamma_{c_i}(I) := P[p_2 \in I_{c_i} \mid p_1 \in I_{c_i}] \,.$$

Figure 11 demonstrates the idea of the modified autocorrelogram, which stores the computed probability out of an area surrounding the pixel into a histogram. The modified autocorrelogram used in our evaluation estimates the probability $\gamma_{c_i}(I)$ based on eight surrounding pixels inside the mask. In addition, a color histogram in IHLS color space is concatenated to the modified autocorrelogram to improve the performance, yet we will simply refer to this joint descriptor as *correlogram*. Overall a 600-dimensional image descriptor is obtained (see [30]).

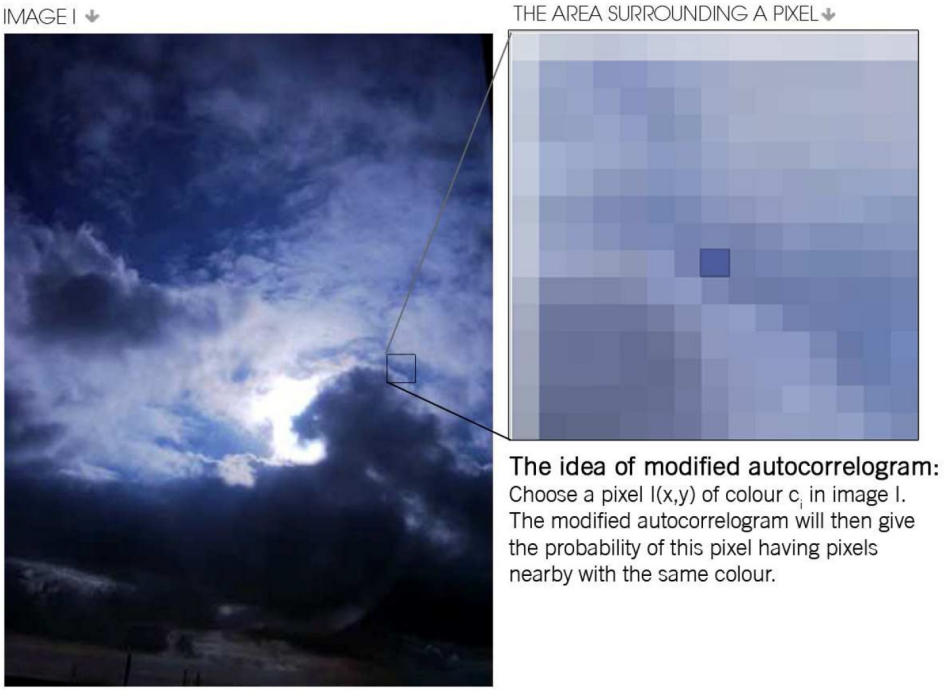


Figure 11: The idea of the modified autocorrelogram (image and caption from [30]).

4.2.5 *Tamura Histograms*

Tamura et al. propose six texture features corresponding to human visual perception: coarseness, contrast, directionality, line-likeness, regularity, and roughness (see [54]).

From experiments testing the significance of these features with respect to human perception, it was concluded that the first three features are very important. Thus, in our experiments coarseness, contrast, and directionality are used to create a histogram describing the texture. Examples to illustrate the meaning of these features are given in Figure 12. Histograms of these features are often used to describe texture properties, for example in the QBIC system (see [20, 23]). In the following we will refer to them as *Tamura Histograms*.

(a) (b) (c) (d) (e) (f)

Figure 12: Example images for texture properties: a) high coarseness b) low coarseness c) high contrast d) low contrast e) directed f) not directed (images and caption from [19]).

4.2.6 *Visual Word Representation*

Visual word representations are a standard feature in object recognition and concept detection and are usually based on local SIFT features introduced by Lowe (see [40]). After these local SIFT features or *patches* have been extracted, they are clustered to form a codebook of *visual words*. Each image is then represented by a vector containing the quantities of appearance of each visual word in the image (similar to the bag-of-words representation known from text retrieval). Because these representations are usually very high-dimensional, a dimensionality reduction using *probabilistic Latent Semantic Analysis* (pLSA) is performed as proposed by Sivic et al. (see [50]). While this visual word representation often outperforms other features, it is computationally expensive, because hundreds or thousands of patches need to be extracted for each image, which then need to be clustered in a preprocessing step to form the codebook.

4.3 EXPERIMENT 1 - COMPARISON OF METHODS AND FEATURES

An experiment is conducted to address the following two questions. First, how does `k-means` perform in comparison to hierarchical agglomerative approaches? And second, which features are suitable for the image clustering task and should be focused on in the next experiments?

Therefore, all non-hierarchical (or flat) clustering techniques are compared for all described features on all datasets except youtube-large which will only be used at the end for the large-scale experiment.

Technical Details

As introduced in Subsection 4.2.4, *correlogram* refers the compound feature consisting of the modified autocorrelogram and a color histogram in IHLS color space. *chist6* means that each axis in RGB color space is quantized into six bins, the same holds for the Tamura Histogram. *viswords2000* refers to the visual word representation using a total of 2000 visual words. Similarly, *viswordsplsa64* refers to the same visual word representation, but with an additional pLSA dimensionality reduction to 64 dimensions (see Subsection 4.2.6).

The conditional entropies H(Cluster | Class) and H(Class | Cluster) refer to the two parts introduced in section 2.2, which are combined to the V-Measure. In this thesis a beta value of 1.0 is always used for the V-Measure. Note that all measures in the left column should be high, whereas the entropy-based measures in the right column should be low.

`k-means` and online `k-means` are compared to hierarchical agglomerative approaches with different linkage methods and different distance measures, which are referred to as "hierarchical *linkage/distance*" in the plots. For the hierarchical agglomerative approaches the *pyClustering* library by De Hoon et al. is used (see [17]). Table 3 gives an overview of the evaluated linkage methods and the distance measures used. Note that the plots only contain complete and average linkage approaches, since single linkage produces even more unbalanced clusters (which are definitely not suitable for image clustering) and centroid-linkage is computationally even more expensive while producing clusters of inferior quality.

The number of clusters was set to the true number of clusters given by the ground-truth partitioning of the datasets. For the hierarchical agglomerative approaches the resulting tree (or dendrogram) is cut at a specific level, so that the desired number of clusters is obtained. For each `k-means` clustering three runs are allowed and the result with the lowest squared error measure is picked.

A sufficiently large maximum number of iterations () was used to allow the algorithm to converge to a local minimum.

Discussion

The results of Experiment 1 can be found in Figures 13, 14, 15, 16 and 17. Obviously, the datasets offer differently demanding tasks. For already mentioned reasons, performance is low especially for the youtube dataset. In contrast, the same methods perform very good on the netclean dataset with up to 90% V-Measure and a Weighted Purity of over 70%, meaning that on average 7 out of 10 images of the same series end up in the same cluster.

From the difference between Purity and Weighted Purity it can be concluded that Purity is not suitable to evaluate highly unbalanced clusterings as produced by the

LINKAGE METHOD	ABBREVIATION
Single-linkage	's'
Maximum- (or complete-) linkage	'm'
Average-linkage	'a'
Centroid-linkage	'c'
DISTANCE MEASURE	**ABBREVIATION**
Pearson correlation coefficient	'c'
Absolute value of the Pearson correlation coefficient	'a'
Uncentered Pearson correlation (equivalent to the cosine of the angle between two data vectors)	'u'
Absolute uncentered Pearson correlation	'x'
Euclidean distance	'e'
City-block or Manhattan distance	'b'

Table 3: An Overview of all linkage methods and distance measures used for hierarchical agglomerative clustering (see [17]).

hierarchical agglomerative approaches as discussed in section 2.2. Average linkage approaches tend to achieve a very high Purity and very low H(Cluster | Class), because they usually produce one giant cluster containing almost all images and multiple very pure but very small clusters. This can also be seen in the high H(Class | Cluster) value, because this giant cluster cannot be homogeneous.

From the plots it can be observed that the correlogram feature consistently performs superior in terms of Weighted Purity and V-Measure for `k-means` and maximum linkage methods. It is only outperformed by the higher dimensional color histogram and the much more complex *viswordsplsa64* feature on the youtube dataset. On all other datasets the correlogram feature performs 20%-43% better for Weighted Purity and 8%-67% better in terms of V-Measure than the second best feature.

As a key result, `k-means` proves to be suitable for image clustering outperforming second best methods on 3 of 5 datasets by 22%-25% and performing only slightly worse on the netclean dataset (-3%) and the youtube dataset (-9%) compared to the best method.

It can be concluded that `k-means` performs surprisingly well compared to the computationally more expensive hierarchical agglomerative methods, which largely

suffer from producing unbalanced structures. Thus, k-means can definitely be deemed suitable for image clustering. Also correlograms outperformed other single features in most cases. Therefore, only correlogram features are used in the next experiments. Note, however, that the overall performance will likely improve when using an adequate combination of multiple features (see [11, 36]).

4.4 EXPERIMENT 2 - HIERARCHICAL CLUSTERING

One major drawback of top-down clustering approaches is that images (or data points in general) which are placed in undesirable clusters are constrained to remain in that branch of the tree (see [11]). The second conducted experiment addresses this issue, and also other related questions: How much cluster quality is lost with the hierarchical top-down approach of h-k-means compared to flat k-means? How does h-k-means performance compare to hierarchical agglomerative methods? Finally, do different branch factors for h-k-means have an impact on performance?

Technical Details

In reference to the first experiment, only correlogram features are used for this experiment and evaluation is limited to Weighted Purity, V-Measure and Entropy. As a representative for hierarchical agglomerative approaches *hierarchical m/b* is chosen, because it was consistently among the best in the first experiment.

The performance of hierarchical clustering is evaluated by measuring performance on every single level of the tree produced by the h-k-means algorithm. Thus, several nested partitions of different numbers of clusters are analyzed. A complete partitioning of the image collection of a specific level of the tree is obtained by taking all nodes of this level and also all leaf nodes which may have already occurred up to this level, which is important to satisfy the completeness requirement. This way h-k-means determines the number of clusters which is used for k-means and the hierarchical agglomerative method, which both are able to produce clusterings for any number of clusters. By applying k-means for specific numbers of cluster of the lower levels of the tree, it can be exactly measured how much performance is lost due to "bad decisions" on upper levels of the hierarchical h-k-means approach. The gray vertical dashed line in every subplot represents the ground-truth number of clusters, where the validity measures can be interpreted most easily. As in the first experiment three runs are allowed for each k-means-based algorithm and the result with the lowest squared error measure is chosen. Again, the number of iterations used is sufficiently large to allow the algorithm to converge to a local minimum.

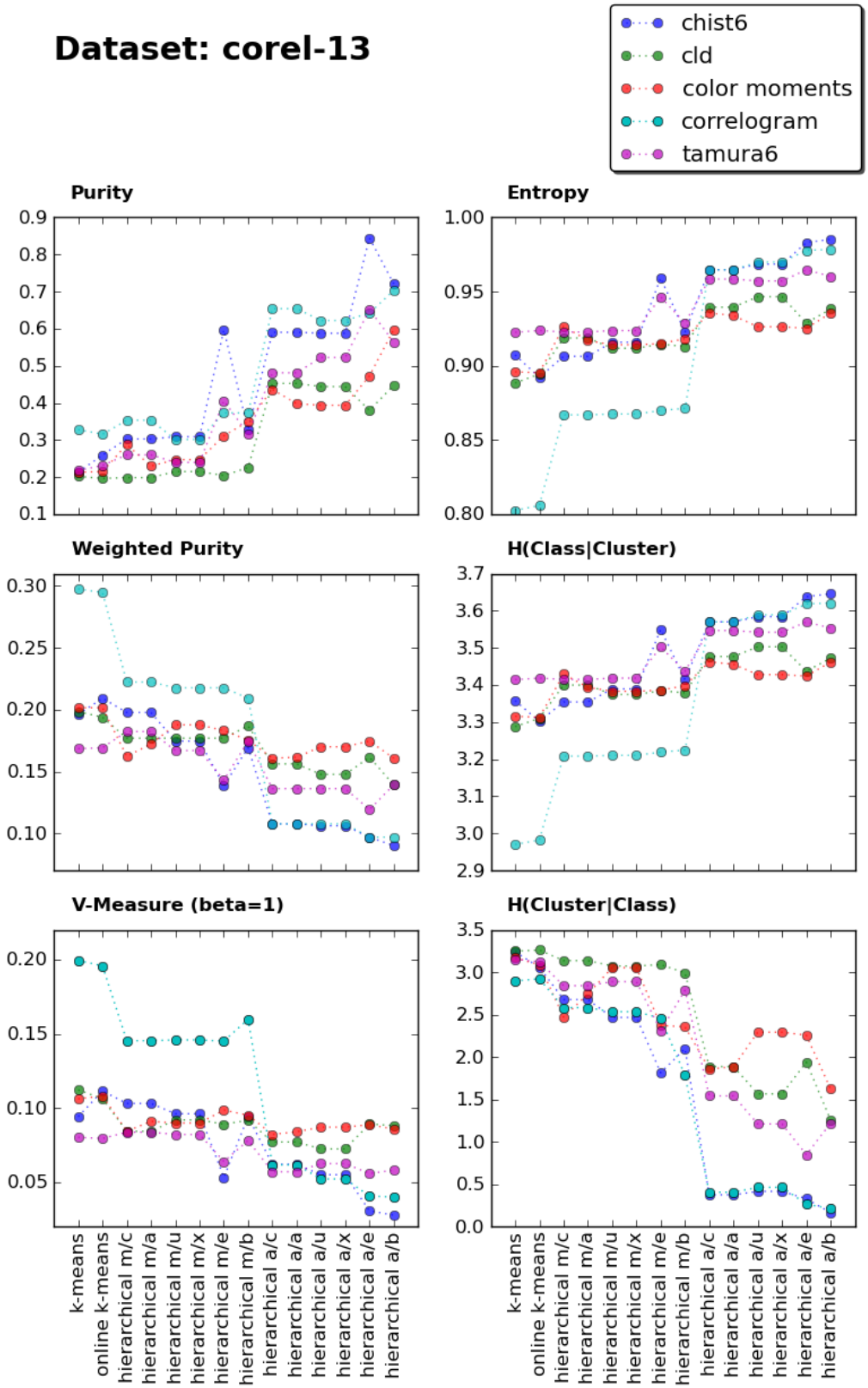


Figure 13: Results of Experiment 1 for the corel-13 dataset. In terms of Weighted Purity and V-Measure `k-means` outperforms other clustering techniques.

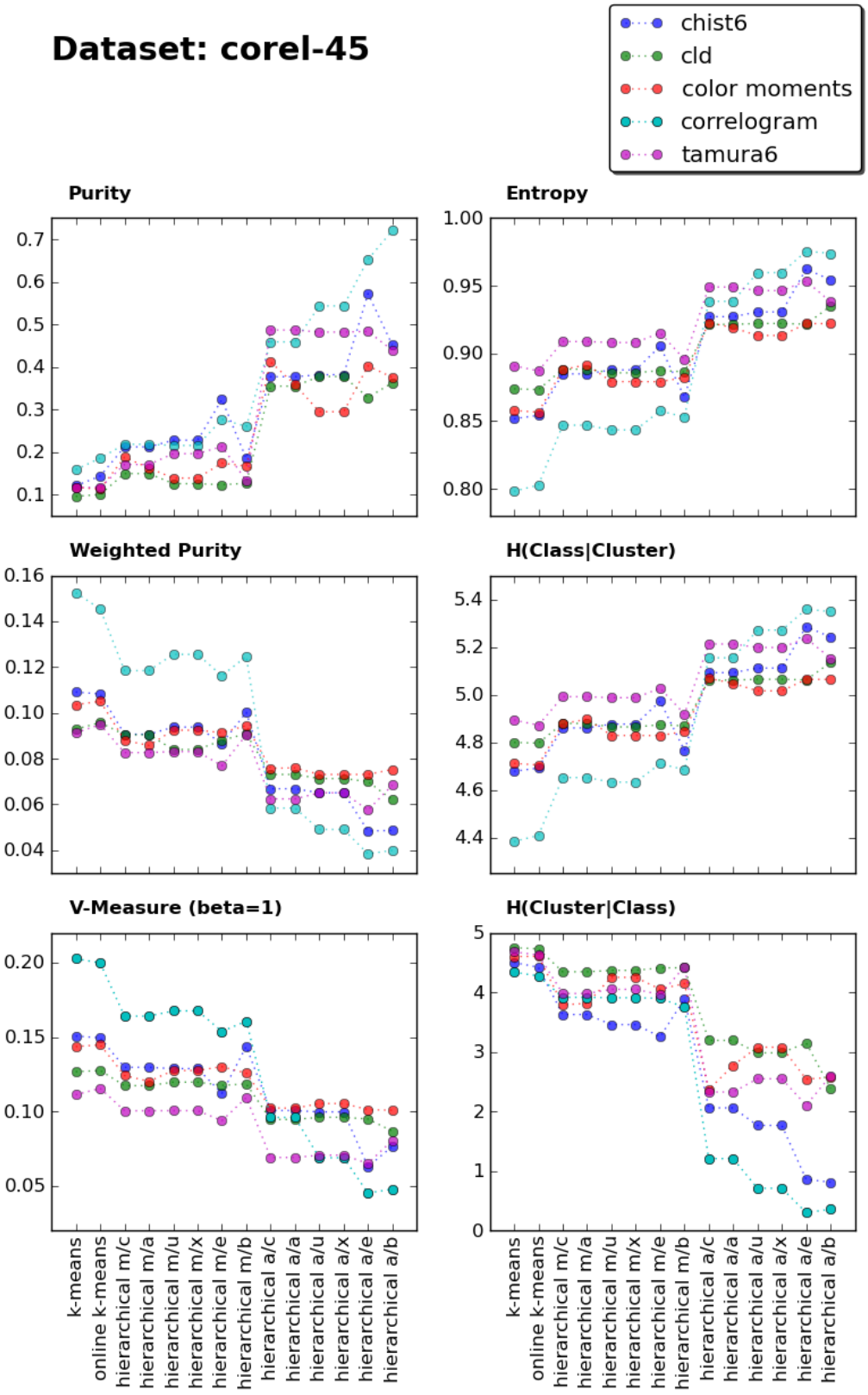


Figure 14: Results of Experiment 1 for the corel-45 dataset. Since the clusters obtained of hierarchical agglomerative methods are very unbalanced, Purity completely distorts the results. H(Class | Cluster) is high for average linkage methods, because one giant cluster contains almost all images.

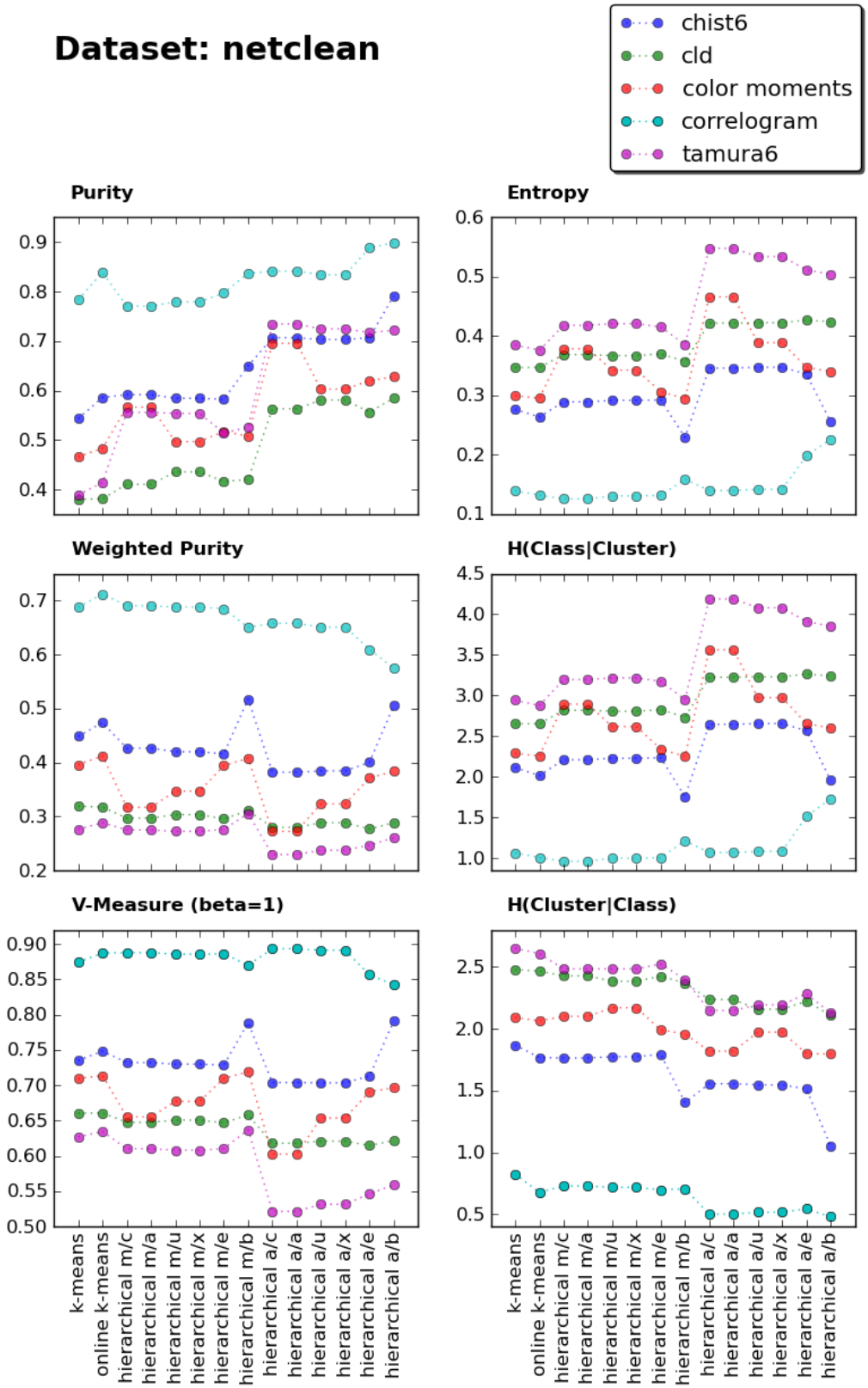


Figure 15: Results of Experiment 1 for the netclean dataset. On this dataset the correlogram feature performs consistently well for all clustering methods. However, it also accomplishes good results on different datasets.

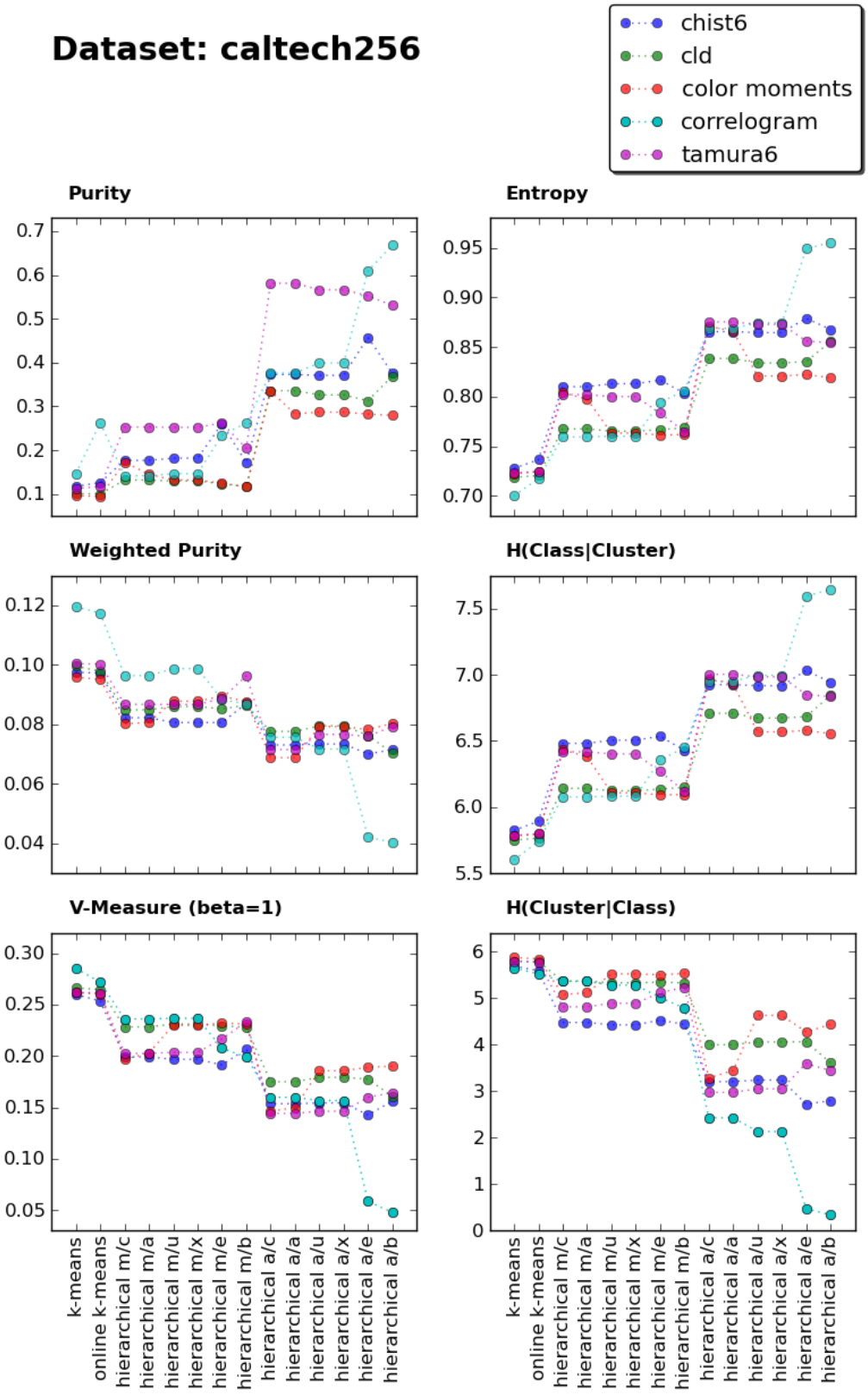


Figure 16: Results of Experiment 1 for the caltech256 dataset. `k-means` is suitable for scalable image clustering, because it performs best in combination with correlogram features for almost all datasets.

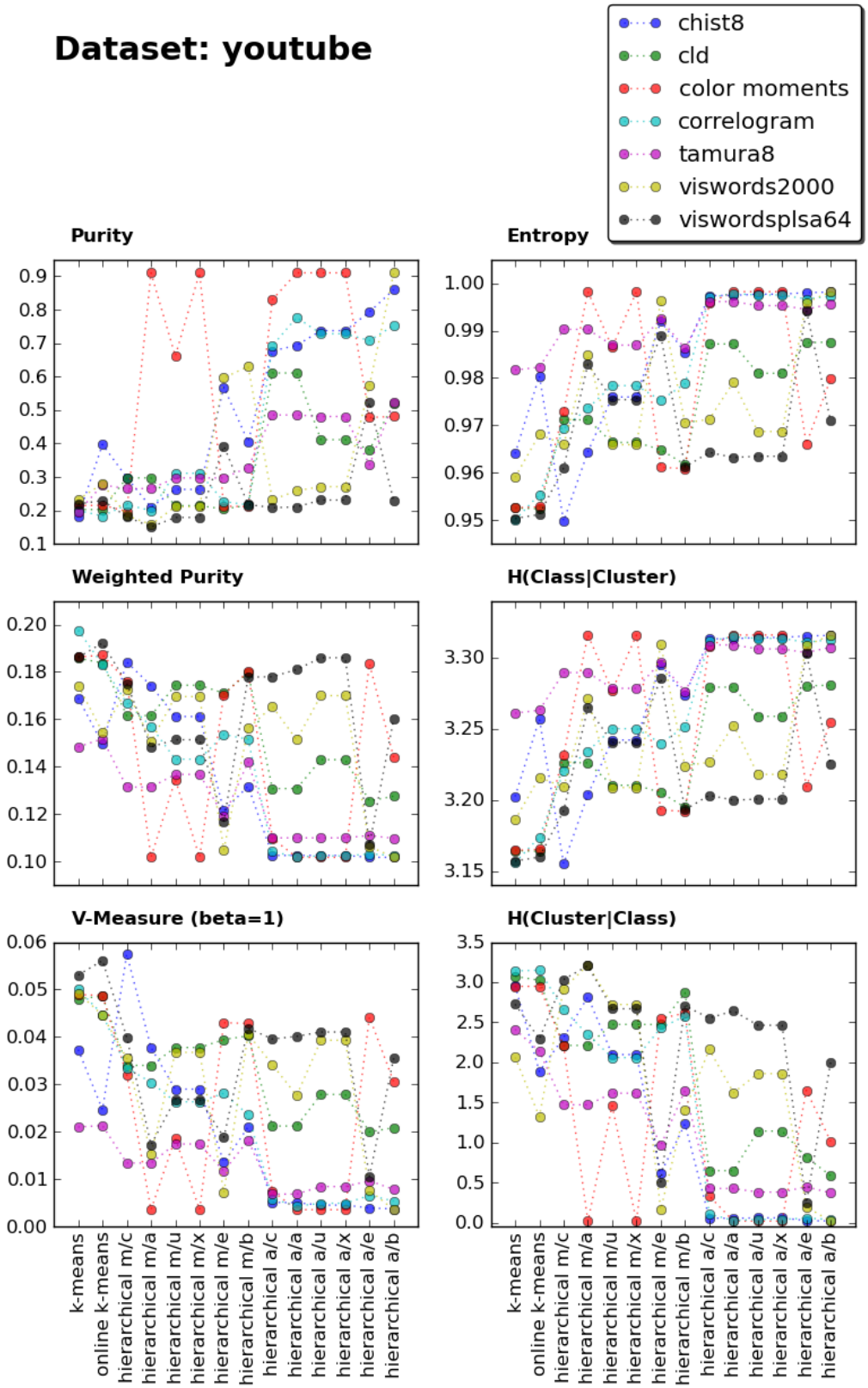


Figure 17: Results of Experiment 1 for the youtube dataset. The overall performance is low due to the high intra-class variance of the youtube dataset. The correlogram feature is only outperformed by *chist8* and *viswordsplsa64*.

Discussion

Figures 18, 19, 20, 21 and 22 show the results of this experiment. The overall behavior is coherent for all used validity measures and unsurprisingly `k-means` is consistently superior to `h-k-means`.

One pattern is noticeable through all datasets. In general `h-k-means` as a top-down approach performs better for a small number of clusters, the left side of all plots, than the hierarchical agglomerative method, which works bottom-up. Accordingly, this is reversed on the right side of the plots, where we have usually thousands of clusters. In between, the point where agglomerative clustering starts to perform superior depends on the dataset. This behavior can be explained by the constraint that images which are placed in undesirable clusters remain in that branch of the tree. These errors pile up over multiple iterations and lead to the described pattern.

The behavior on the netclean dataset differs from all other datasets. This is due to the comparatively small cluster sizes of this dataset, which are obtained only at the lower levels of the tree. On these lower levels the agglomerative approach and `k-means` clearly but unsurprisingly outperform `h-k-means`. For all other datasets and the ground-truth number of clusters `k-means` works better than `h-k-means` (by 5%-19% for Weighted Purity and 7%-28% for V-Measure), which again outperforms the agglomerative method (by 8%-24% for Weighted Purity and 7%-80% for V-Measure, values for a branch factor of 2). Additionally, on all datasets except netclean the performance methods "converge" to approx. the same value.

Another observation is that there is virtually no difference in performance for different branching factors. The impact of different branch factors will be more explicitly evaluated for the `hbo-k-means` algorithm in section 4.6.

In conclusion, `h-k-means` performs clearly superior for small numbers of clusters whereas the hierarchical agglomerative method outperforms the top-down approach on lower levels of the tree. For the ground-truth number of clusters it can be concluded to lose performance of about 5%-19% for Weighted Purity and 7%-28% in terms of V-Measure compared to the flat `k-means` approach.

Note that flat `k-means` merely constitutes a control run, since no hierarchical structure is obtained which is needed for browsing. In addition, recall that h-k-means leads to significant speed benefits compared to hierarchical agglomerative approaches. Finally, it can be seen from the results that the branch factor seems to have almost no impact on performance.

4.5 EXPERIMENT 3 - THE IMPACT OF BALANCING

As already described in Section 3.2, balancing is a important requirement for the task of image clustering for browsing applications. From Experiment 1 it is evident that while hierarchical agglomerative approaches produce very unbalanced structures,

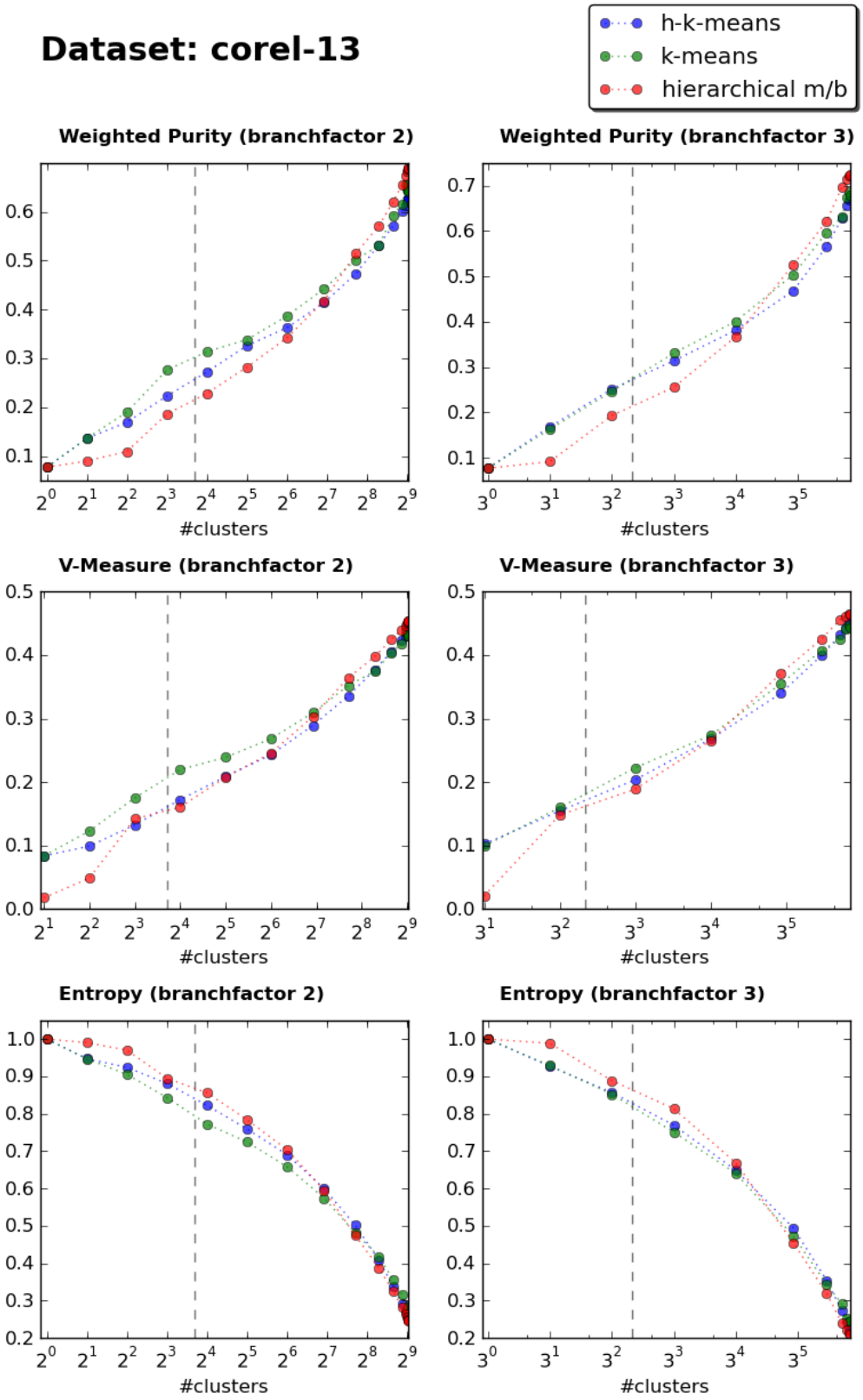


Figure 18: Results of Experiment 2 for the corel-13 dataset. It can be observed that top-down methods are the best for small numbers of clusters (left) and bottom-up methods for large numbers (right).

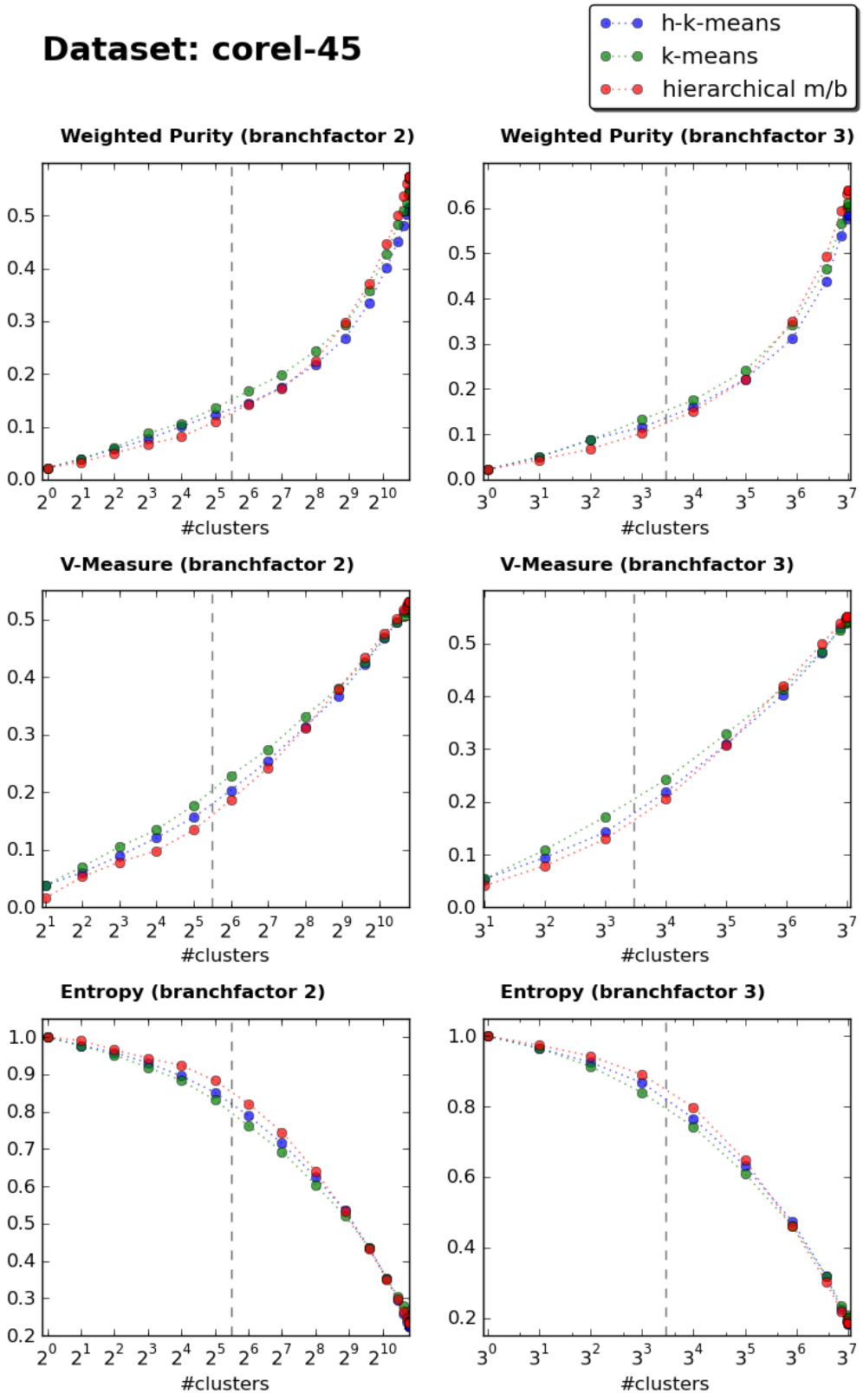


Figure 19: Results of Experiment 2 for the corel-45 dataset. The overall behavior is coherent for all used validity measures, and unsurprisingly `k-means` is consistently superior to `h-k-means`.

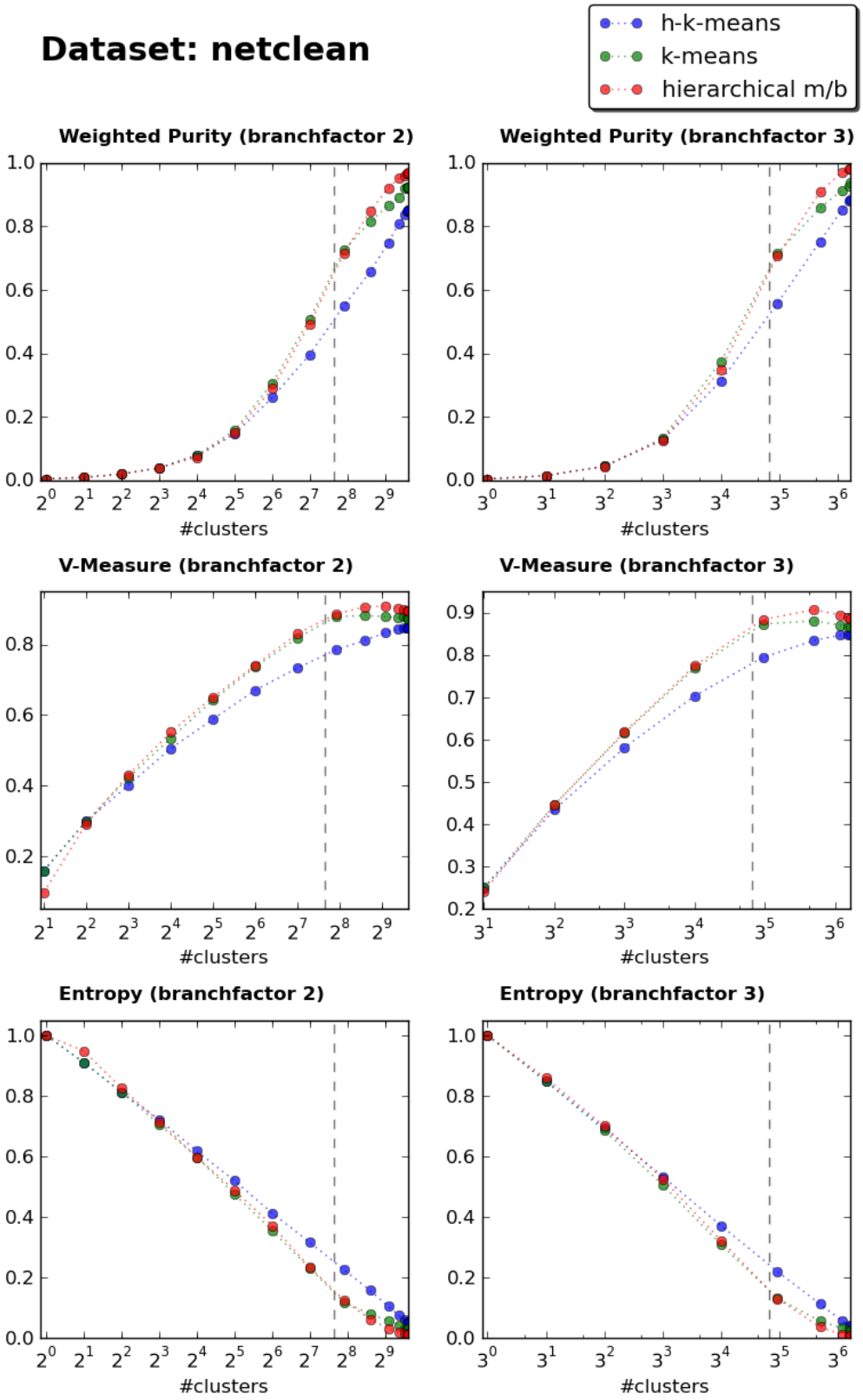


Figure 20: Results of Experiment 2 for the netclean dataset. Due to the comparatively small cluster sizes (of the ground truth partitioning) of the netclean dataset the behavior on this dataset differs from all other datasets.

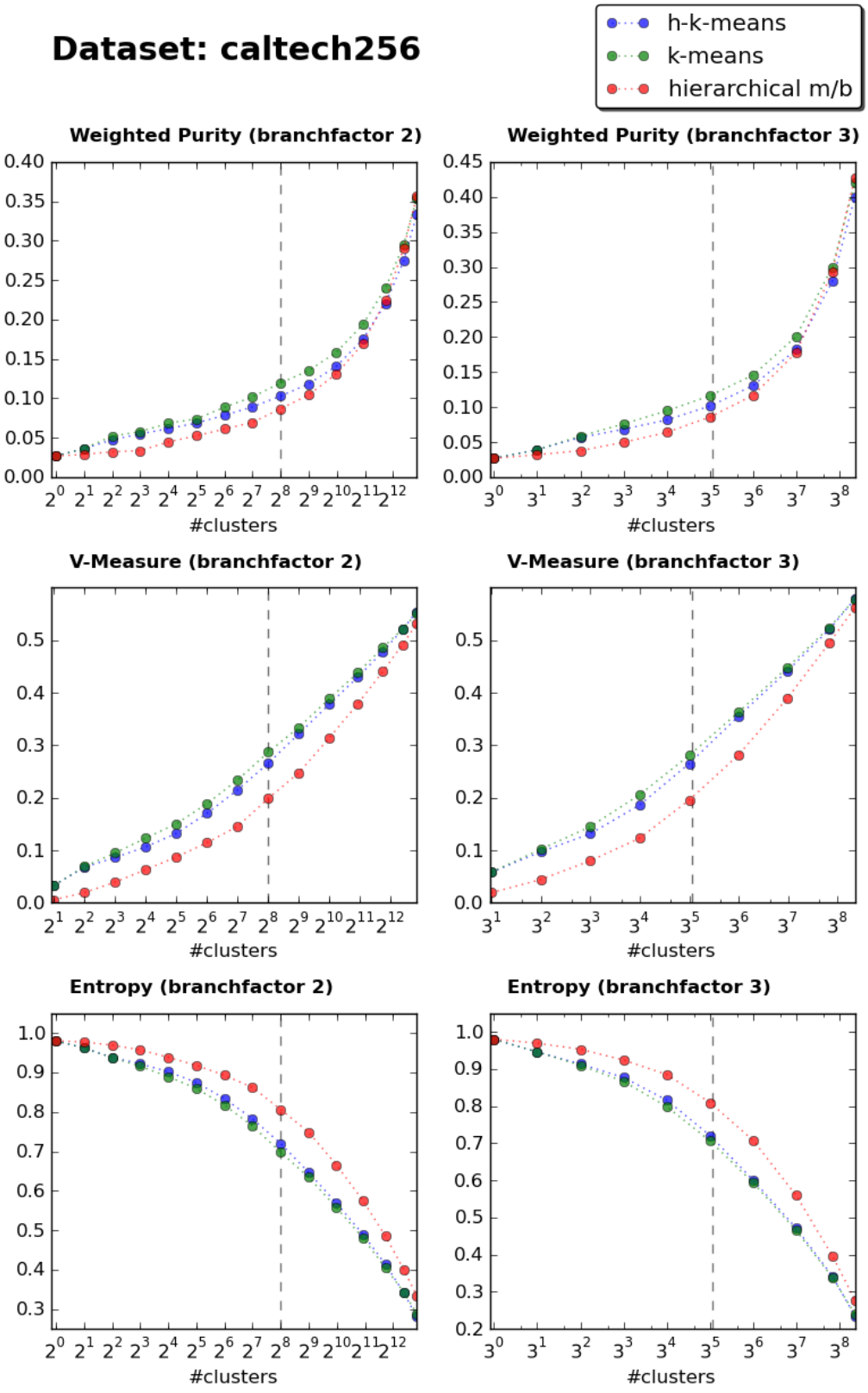


Figure 21: Results of Experiment 2 for the caltech-256 dataset. Different branch factors tend to have virtually no impact on performance.

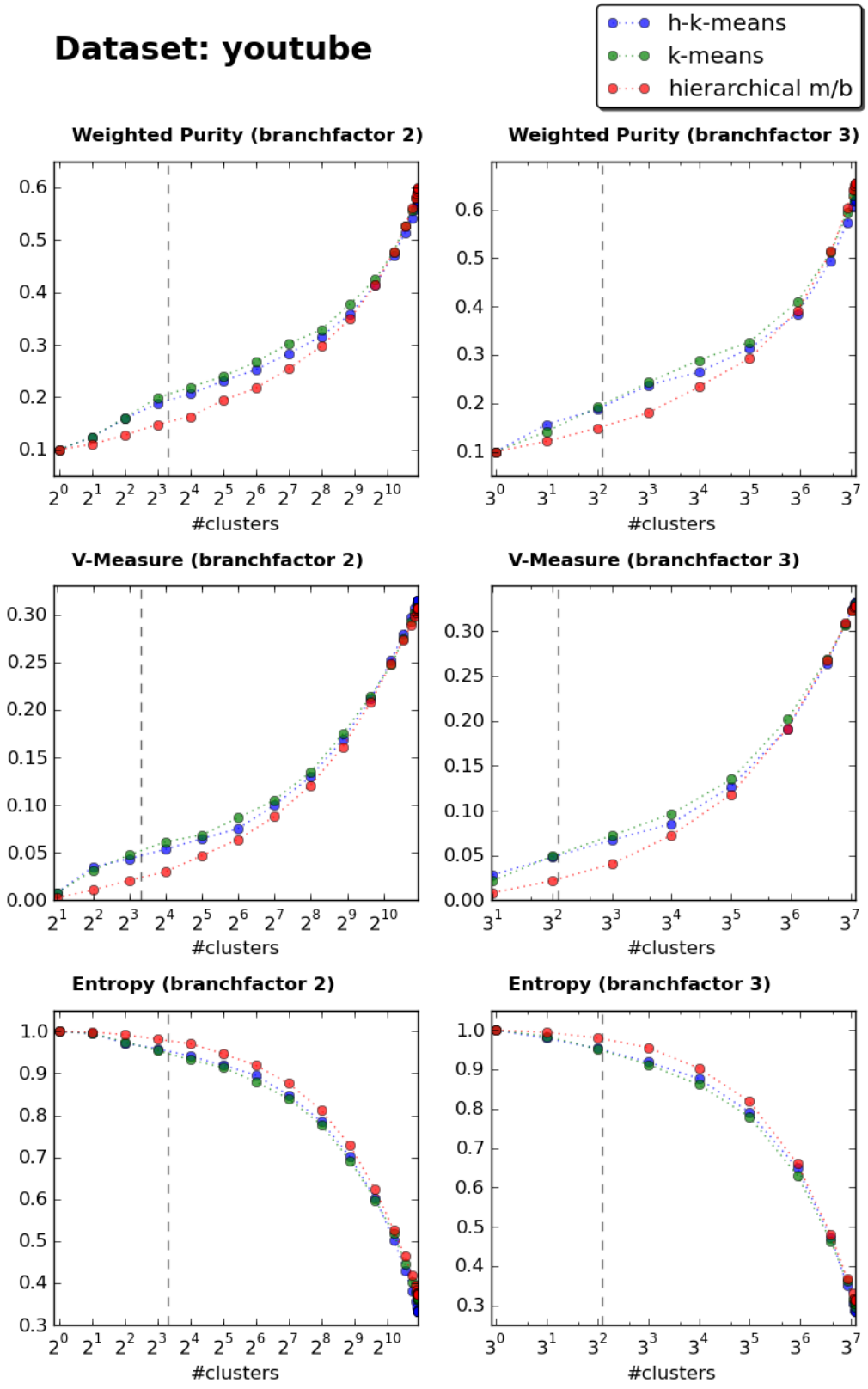


Figure 22: Results of Experiment 2 for the youtube dataset. It can be seen that the performance for all methods "converges" to approx. the same value for the lower levels of the tree.

this is also to a lesser extent the case for the `k-means` algorithm. However, especially when initialized poorly `k-means` will produce very small outlier clusters. It has been introduced that balancing approaches can provide a form of regularization to avoid low-quality local minima stemming from poor initializations. In this experiment this is investigated quantitatively. More precisely the following questions are asked: "What is the impact of the proposed balancing strategy both on the distribution of cluster sizes and the quality of the obtained clusters?", "How does the proposed `bo-k-means` algorithm compare to other balancing approaches?" and "Does the balancing approach generalize well with respect to different features?".

Technical Details

In this experiment the balancing impact on flat clustering methods is investigated, before proceeding to hierarchical approaches in the next experiment. Therefore, the proposed `bo-k-means` algorithm is used (see Algorithm 4). Since results of `k-means`-based algorithms depend on the quality of the initialization (especially because it is done randomly) ten runs for each value of β are allowed. The colored points in the plots depict the average value while the error bars represent the standard deviation. In contrast to the previous experiments each of these 10 runs is in fact one single run and is not initialized multiple times to pick the best result. The maximum number of iterations was set to 50 full runs over the dataset (called *epochs*), but in most cases the algorithm converged significantly earlier (a `convergence_threshold` of 2% was used).

Now, we introduce three measures to evaluate balancing (see [4]). The *standard deviation in cluster sizes* (SDCS) often helps in understanding the balancing behavior of a clustering algorithm. In addition, it is useful to know whether an algorithm produces very small or very big clusters. To quantify this behavior, we use the ratio of the minimum cluster size to the expected cluster size under perfect balancing. We refer to this as *ratio of minimum to expected* (RMinE). Analogously, we define the *ratio of maximum to expected* (RMaxE). Formally:

$$\begin{aligned}
\text{SDCS} &= [\frac{1}{m}\sum_{j=1}^{m}(n_j - \frac{N}{m})^2]^{1/2} \,, \\
\text{RMinE} &= \min_i(n_i)/(\frac{N}{m}) \,, \\
\text{RMaxE} &= \max_i(n_i)/(\frac{N}{m}) \,.
\end{aligned}$$

In this experiment, Quality is measured by V-Measure and Weighted Purity, and balancing by SDCS and minimal and maximal cluster size, which are proportional to RMinE and RMaxE, as introduced above. Note that the Standard Deviation in Cluster Sizes has its own y-axis on the right side of each subplot.

Discussion

Figures 23 and 24 illustrate the results. Recall that through β true feature similarity can be continuously traded off against balancing constraints. A value of 0 implies no balancing at all while a value of 1 means solely relying on cluster sizes. Without balancing very small and very large clusters occur, being strongly dependent on the initialization. Balancing provides a strong regularization effect, which renders the result almost initialization-independent (graphically the error bars disappear). Our proposed balancing approach proves to be highly effective, rapidly forcing all cluster sizes to be (almost) equal.

Balancing never deteriorates the cluster quality right away. Either the quality stagnates up to a large value of β (e.g. on caltech-256) or balancing is able to improve accuracy. This is especially evident on the netclean dataset, where for example Weighted Purity goes up by 33%. Of course, quality quickly decreases for very large values of β, because then the balancing algorithm merely equalizes the cluster sizes regardless of the actual feature similarity.

From the results of the *viswordsplsa64* feature of the youtube dataset it can be concluded that our approach also generalizes to different features. However, the balancing seems to effective more slowly. This is an important result because we need to choose a fixed β value that is likely to perform well for unknown datasets in practice.

Our proposed approach was also compared to different FSCL methods introduced in subsection 3.3.2. Unfortunately, these approaches tended to be too unstable to be suitable for thorough quantitative evaluation in our high-dimensional feature space. Thus, they are not suited for the task of image clustering for browsing (for more detail see subsection 3.3.2).

From the given results it can be concluded that our proposed balancing strategy improves the distribution of cluster sizes strongly by producing more balanced clusters. Additionally, for a wide stable range of the key parameter β our `bo-k-means` method never impairs but in some cases even improves clustering quality. Perhaps more importantly, balanced clustering results tend to be initialization-independent. This is very nice, because the clustering algorithms do not need to be repeated multiple times, resulting in even more efficient image clustering. In addition, our algorithm seems to generalize well to different feature spaces and outperforms other balancing approaches in terms of stability. For subsequent experiments β will be set to 0.4, which resulted in (near-)optimal cluster quality for all evaluated datasets. Especially the results on the netclean dataset show that a much larger value should not be picked. Thus, one could argue that true feature similarity should be weighted 50% more important than balancing constraint in order to optimize the clustering results.

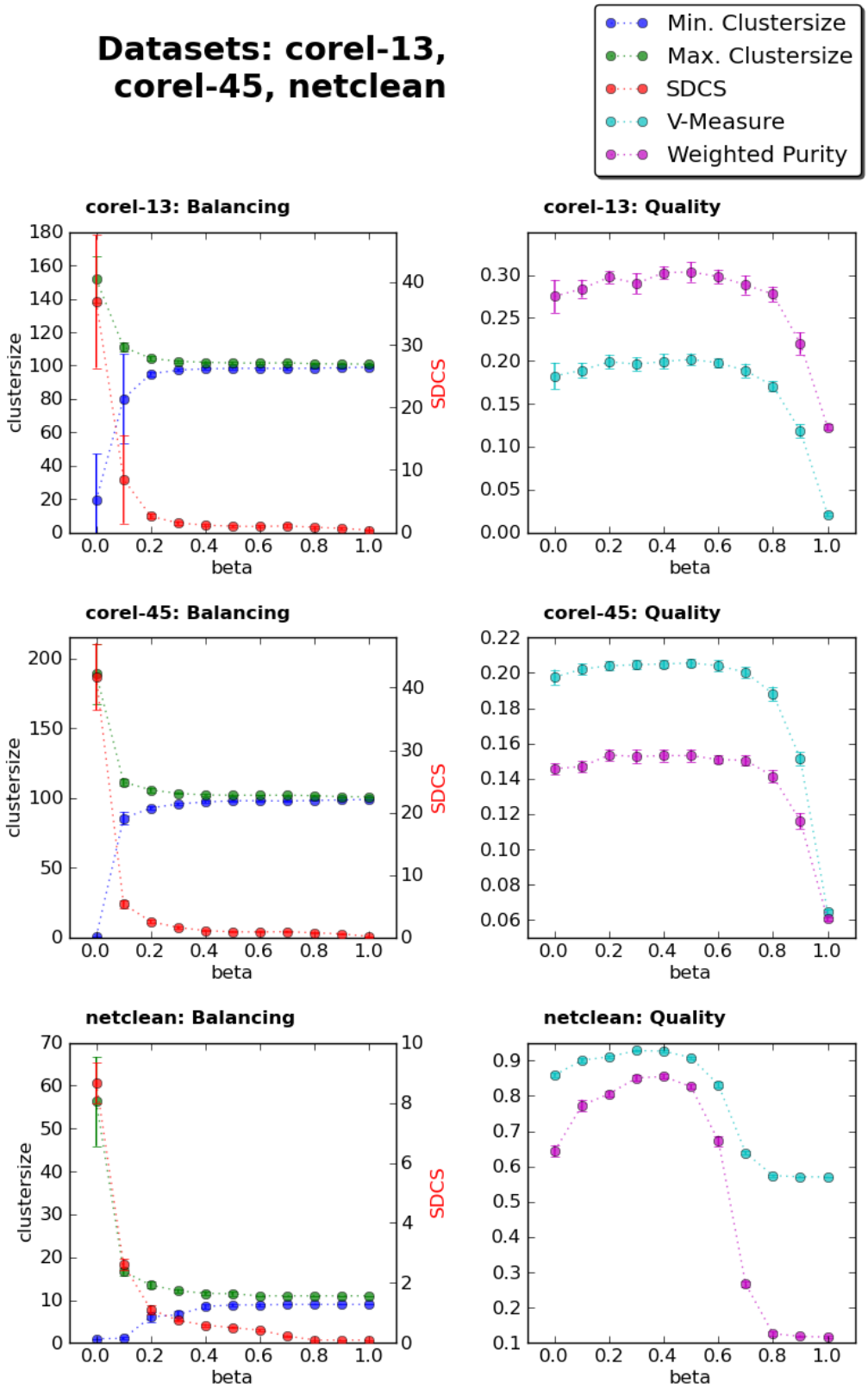


Figure 23: Results of Experiment 3 for the corel-13, corel-45 and the netclean dataset. Balancing never impairs cluster quality but improves it in some cases (for instance on the netclean dataset).

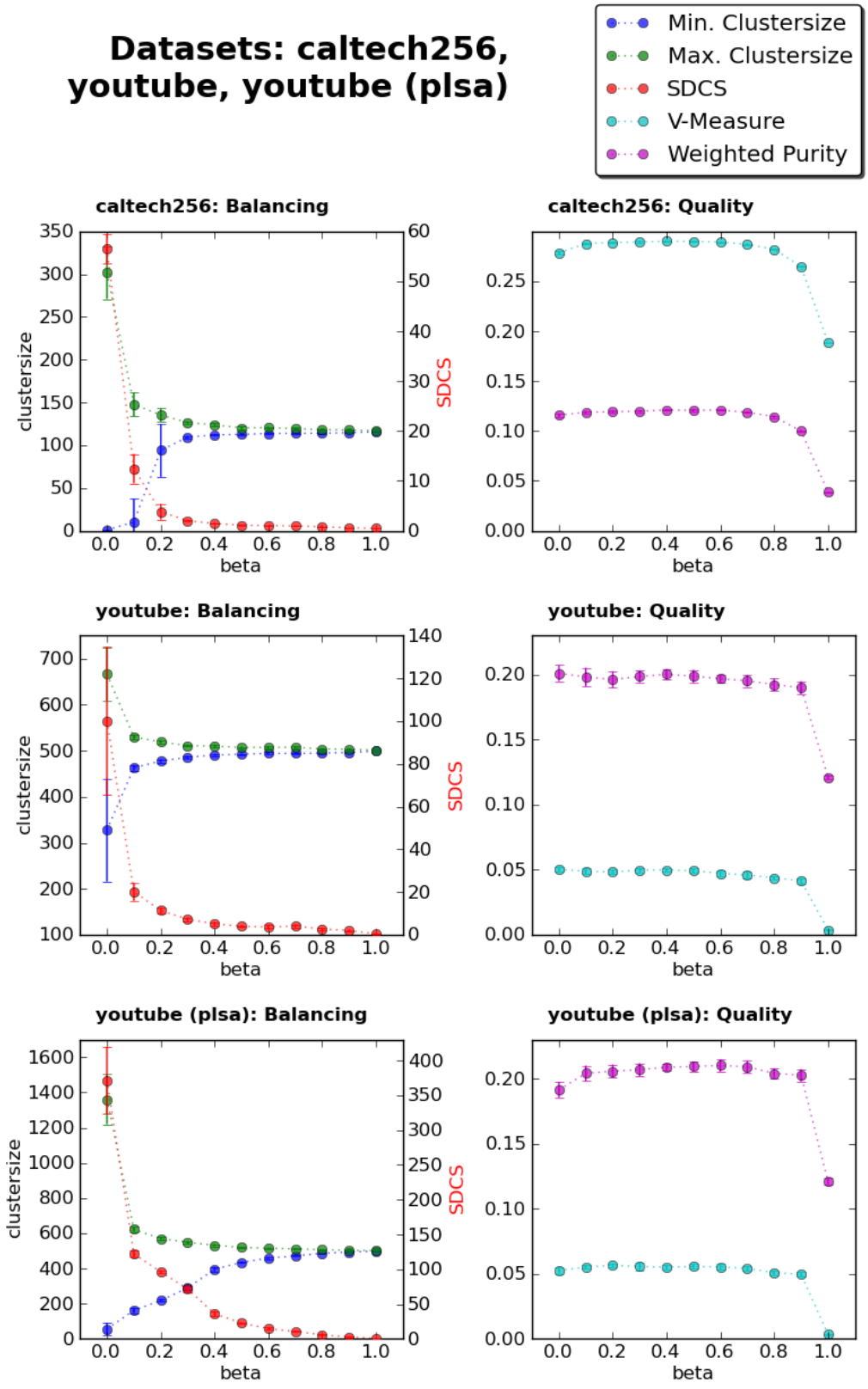


Figure 24: Results of Experiment 3 for the caltech256, and the youtube dataset. For the youtube dataset both correlograms and visual words pLSA features were used to investigate how good the proposed `bo-k-means` algorithm generalizes with respect to different features. Without balancing very small and very large clusters occur, being strongly dependent on the initialization. In contrast, balancing renders the clustering result almost initialization-independent.

4.6 EXPERIMENT 4 - THE IMPACT OF BALANCING ON HIERARCHICAL CLUSTERING

The fourth experiment proceeds with the evaluation of our proposed balancing strategy by investigating its impact on *hierarchical* clustering. We are interested in how our balancing approach impacts the hierarchical `k-means` algorithm, again both balancing- and cluster quality-wise, and whether different branch factors have an impact on our proposed algorithm.

Technical Details

Our proposed `hbo-k-means` algorithm and the regular `h-k-means` algorithm are compared to a hierarchical agglomerative method using average linkage and euclidean distance, and to the balanced agglomerative approach proposed by Borth et al., which basically enhances the agglomerative approach by an additive penalty term to enforce balancing equivalently to our balancing extension of `k-means`. The plots refer to these bottom-up approaches as *hierarchical a/e* and *balanced hierarch. a/e*. Due to the huge memory requirements of the $O(N^2)$ distance matrix, we have not been able to evaluate the last two approaches on the caltech-256 dataset, which contains about 30.000 images. This clearly shows the need for more efficient image clustering approaches.

The clustering quality on different levels of the tree is evaluated as described in Experiment 2 (see Section 4.4). The `k-means`-based methods are initialized only once (again no multiple runs) and 50 epochs are allowed as in the third experiment. For `hbo-k-means` again a `convergence_threshold` of 2% is used.

The plots comparing the branch factors (figures 30, 31 and 32a) only show the performance of our `hbo-k-means` algorithm.

Discussion

The results of the fourth experiment can be found in Figures 25, 26, 27, 28 and 29. Table 4 summarizes the runtime for the different clustering methods on each dataset.

From all results it can be observed that the pattern of top-down methods being superior on the upper levels and bottom-up methods on the lower levels still persists. Balancing especially improves the hierarchical agglomerative approach, causing it to overtake top-down methods quality-wise much earlier. However, the runtime results show that our proposed approach scales much better than hierarchical agglomerative methods. This is particularly evident on the caltech-256 dataset where `hbo-k-means`is about 43 times faster than *hierarchical a/e*.

Our `hbo-k-means` algorithm offers comparable and sometimes slightly superior performance compared to non-balanced `h-k-means` (e.g. on corel-45 or netclean dataset). The approach also captivates in terms of balancing efficiency, driving down

DATASET	#IMAGES	#CLUSTERS	h-k-means	hbo-k-means	*hierarchical a/e*
corel-13	1300	13	4.0	11.0	7.5
corel-45	4500	45	18.3	49.1	122.2
netclean	2000	200	8.2	16.0	15.7
caltech-256	29780	256	398.1	494.8	21411.6
youtube	5000	10	22.2	65.5	160.6

Table 4: The runtime for h-k-means, hbo-k-means and *hierarchical a/e* for each dataset. For the k-means-based methods we used a branch factor of two. Especially the results on the largest dataset, caltech-256, indicate that the proposed approach scales very well.

SDCS to 3%-5% for all datasets except netclean with 14% (measured at the peak at the first tree level for a branch factor of 2 in comparison to h-k-means).

Figures 30, 31 and 32a show, that different branch factors have practically no effect. Even perceived differences as for SDCS on the caltech-256 dataset turn out to be mostly imperceptible (differences of less than 150 in cluster sizes when the average cluster size is about 1000).

It can be concluded that our proposed hbo-k-means algorithm consistently performs comparable or slightly better than h-k-means in terms of cluster quality. In addition, it produces highly balanced clusters of almost the same size on all levels of the tree (SDCS driven down to 3% compared to h-k-means). Taking a closer look at the different branch factors, it can be seen that these have practically no effect on neither cluster quality nor balancing. Therefore, this parameter can be easily changed to meet particular applications needs.

4.7 EXPERIMENT 5 - THE LARGE-SCALE TEST

In Section 3.2 the requirement of scalability was imposed as a key challenge addressed in this thesis. To test whether the hbo-k-means algorithm meets this requirement to a sufficiently large extent, the proposed method was tested on the youtube-large dataset, which contains 716266 images of 225 different semantic concepts. The same experimental setup and parameters as in Experiment 4 (see Section 4.6) were used. The clustering took 4:07 hours using one 1150MHz core of a Quad-Core AMD Opteron(tm) Processor 2356 machine and a maximum of 9.2GB of RAM (mainly due to the high-dimensional features). Considering the huge image collection this duration is small enough to be meaningfully applied in practice. Since a target clustering corresponding to the 225 semantic concepts is available, external cluster validity measures could be applied. The results are shown in Figure 32b.

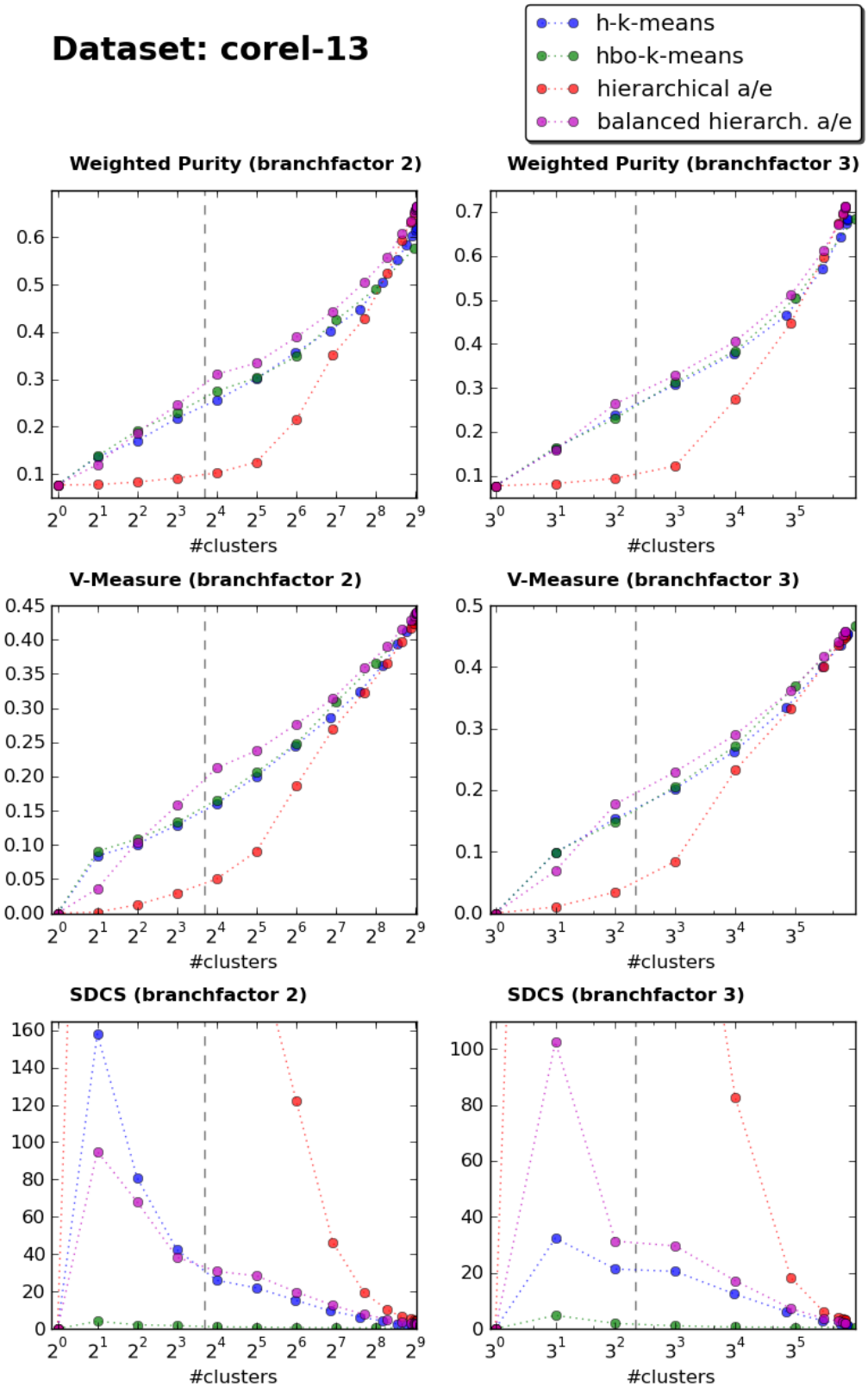


Figure 25: Results of Experiment 4 for the corel-13 dataset. The pattern of top-down methods being superior on the upper levels and bottom-up methods on the lower levels also persists when using balancing.

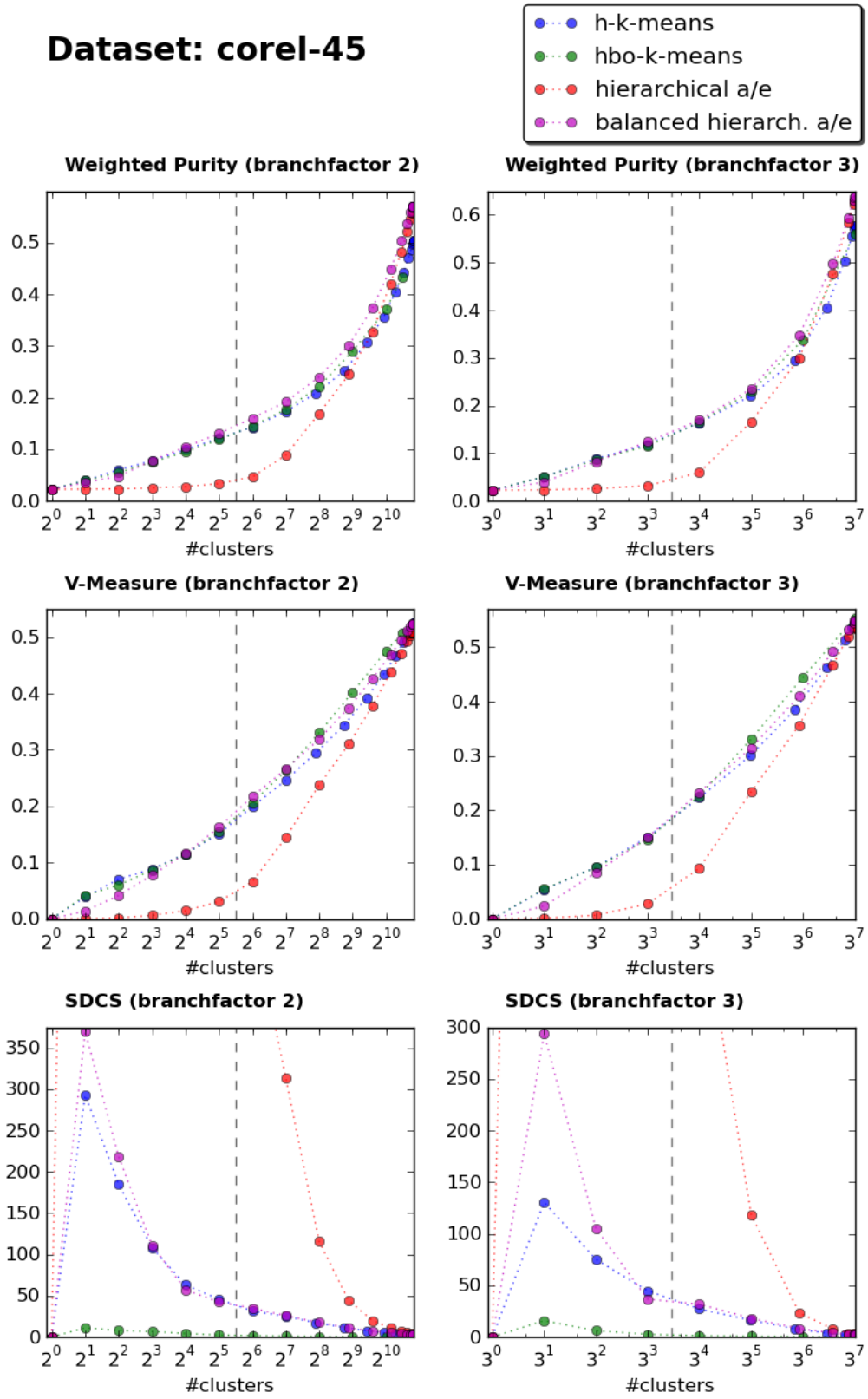


Figure 26: Results of Experiment 4 for the corel-45 dataset. Balancing strongly impacts the hierarchical agglomerative approach, causing it to overtake top-down methods quality-wise much earlier.

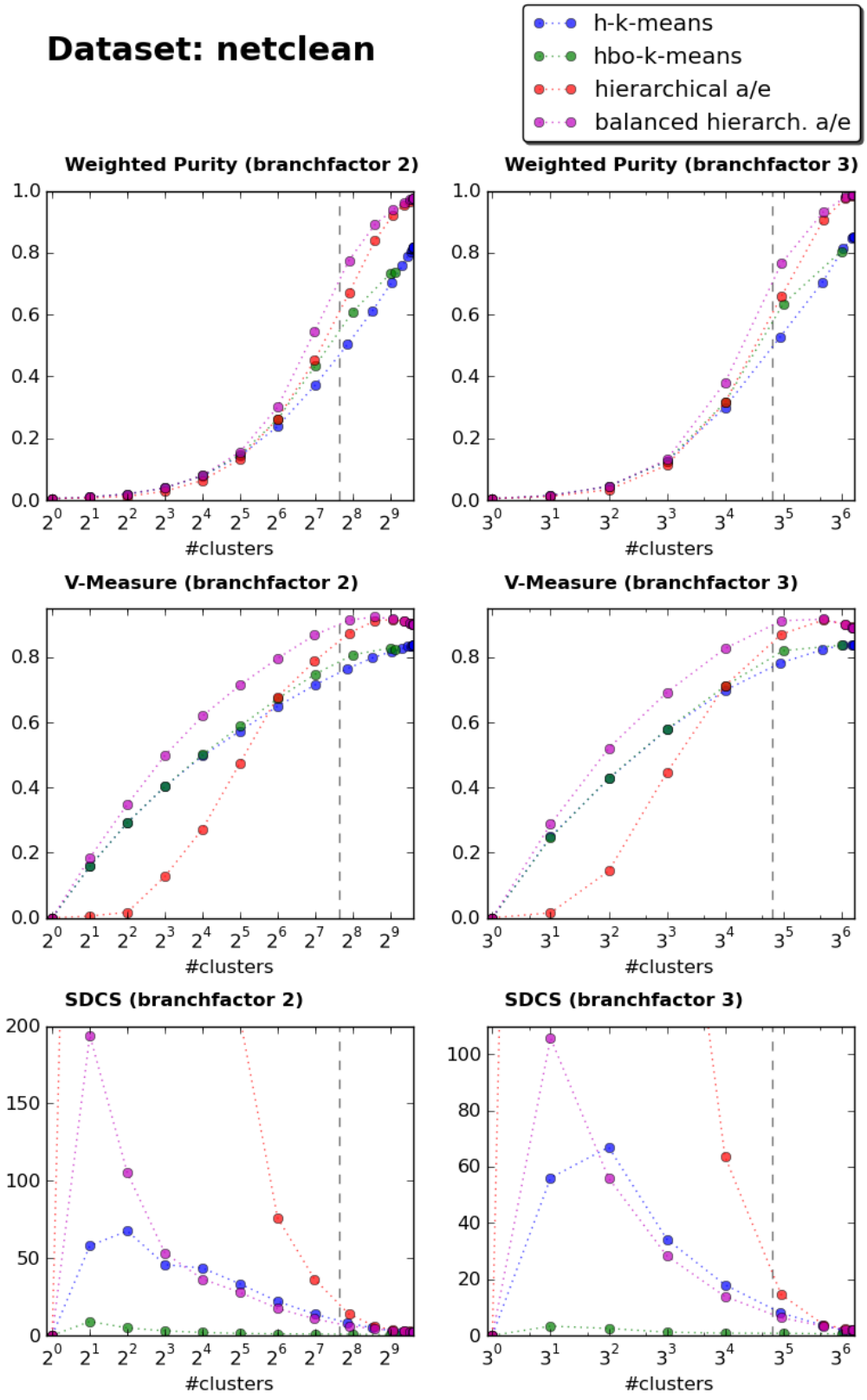


Figure 27: Results of Experiment 4 for the netclean dataset. The `hbo-k-means` algorithm offers comparable and some time slightly superior performance compared to non-balanced `h-k-means`.

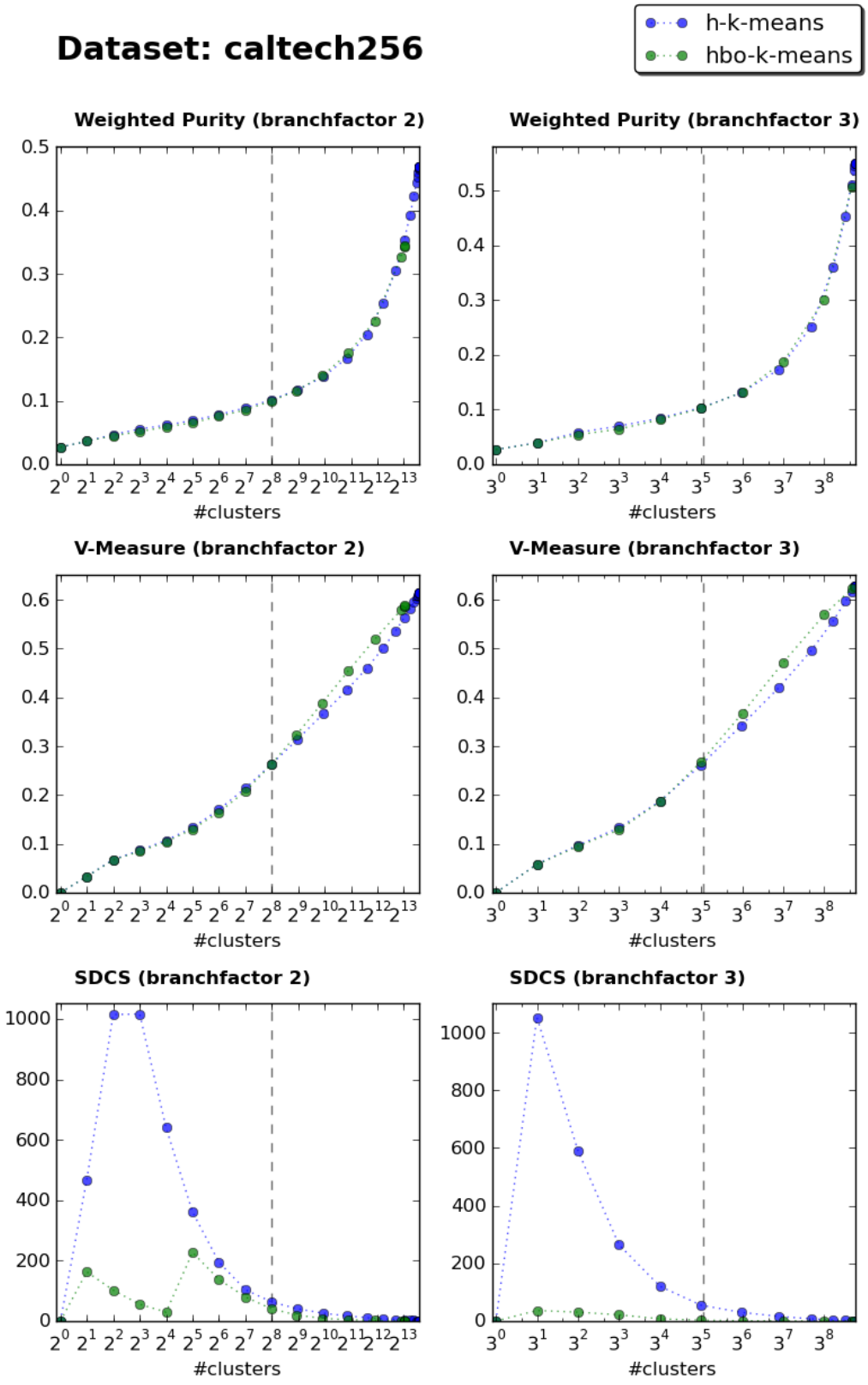


Figure 28: Results of Experiment 4 for the caltech-256 dataset. We have not been able to evaluate hierarchical agglomerative approaches on the caltech-256 dataset due to the $O(N^2)$ memory complexity. This clearly shows the need for more efficient image clustering approaches.

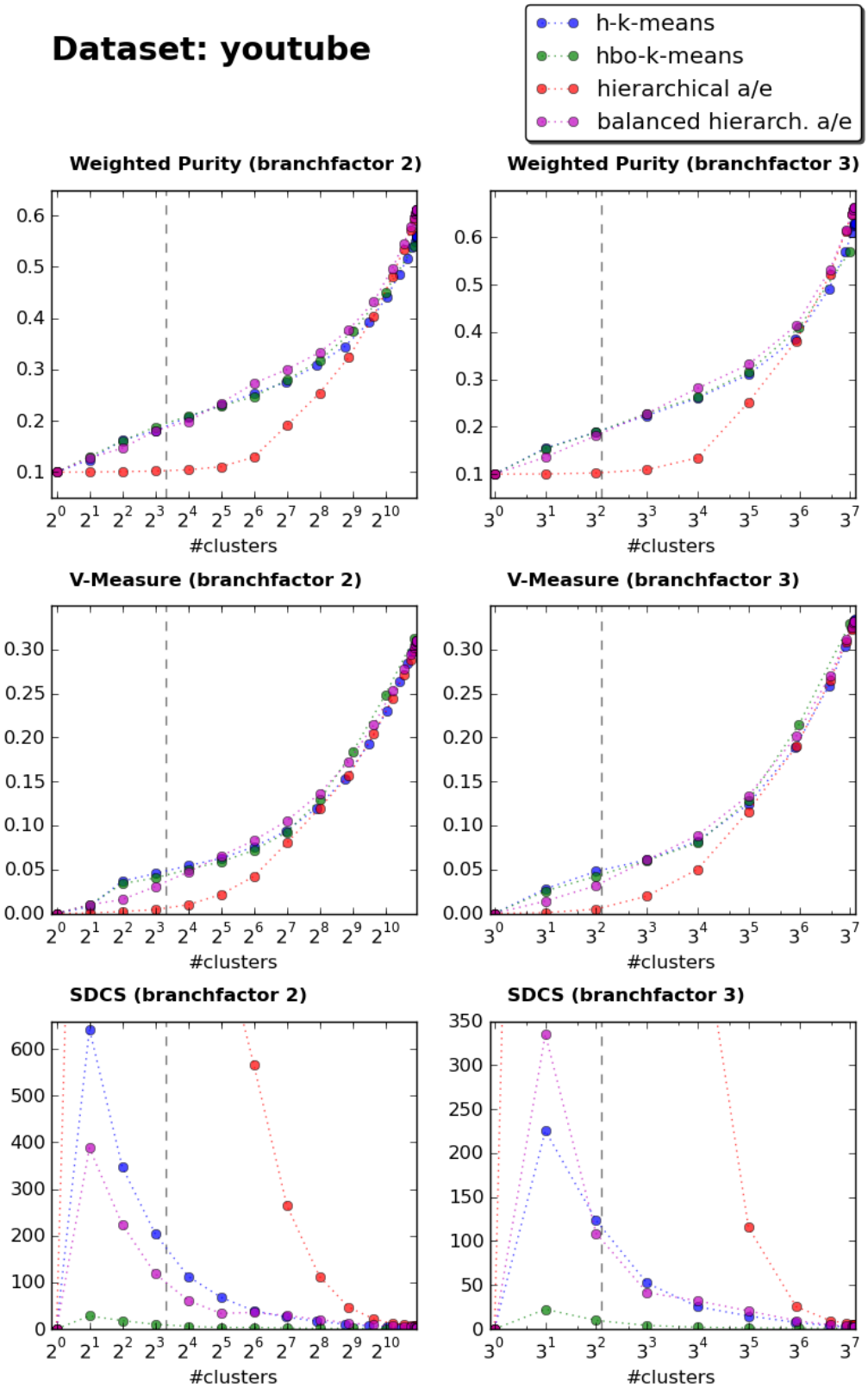


Figure 29: Results of Experiment 4 for the youtube dataset. `hbo-k-means` effectively produces highly balanced clusters, driving down SDCS to 3%-5% in comparison to `h-k-means`.

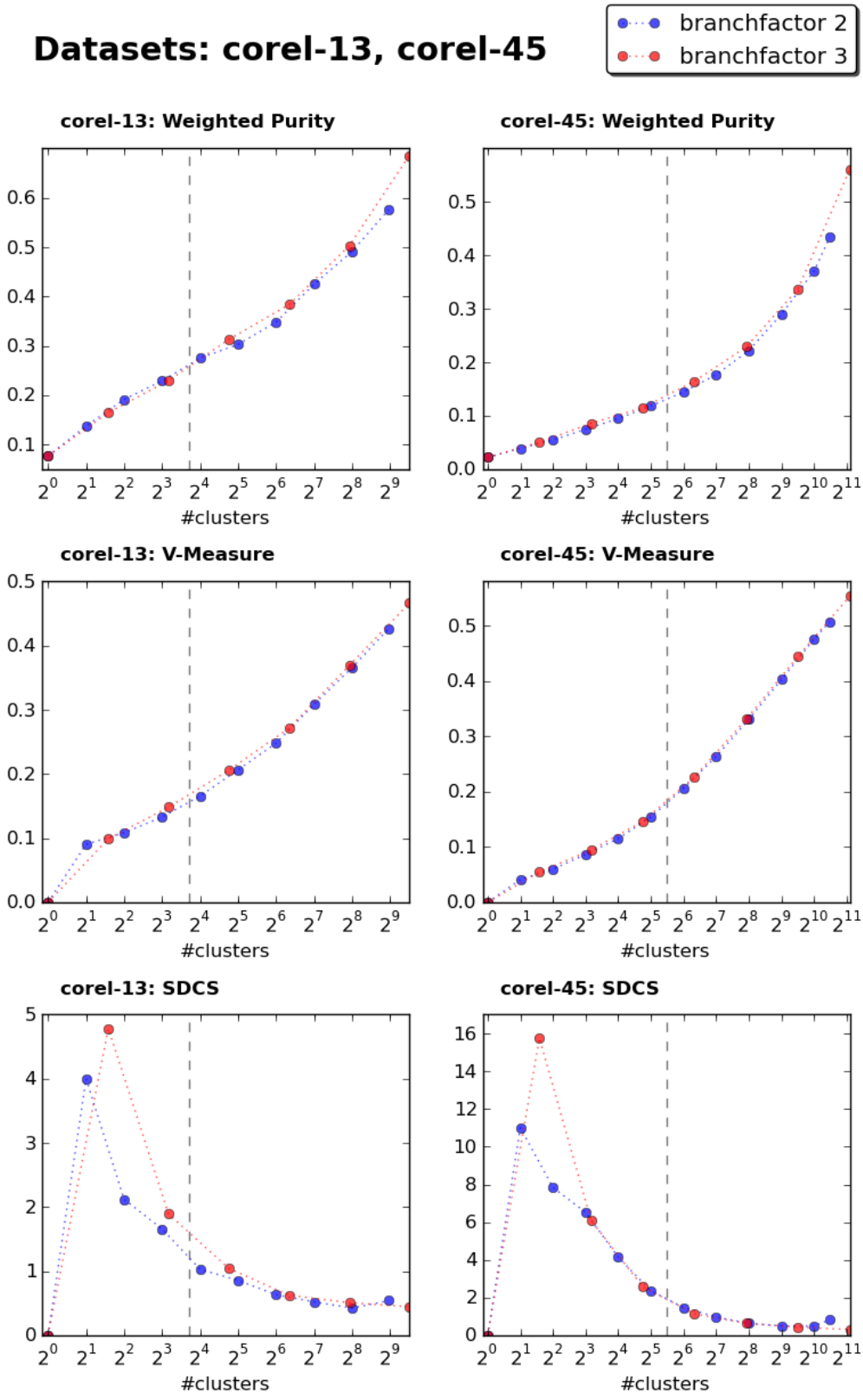


Figure 30: Results of Experiment 4 comparing different branchfactors for the corel-13 and the corel-45 dataset. The branch factor can easily be changed to meet particular applications needs, because it has practically no effect on neither cluster quality nor balancing.

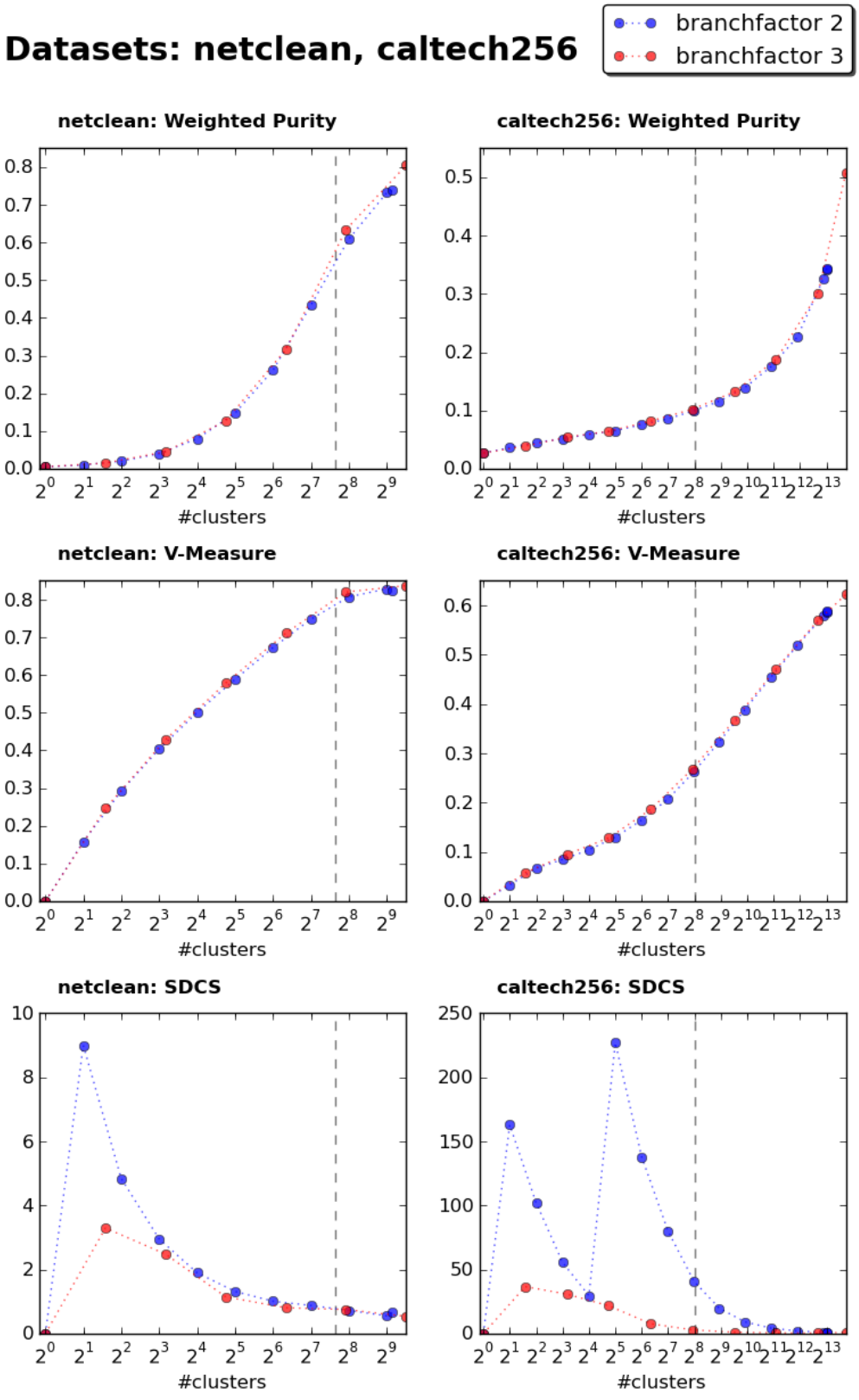
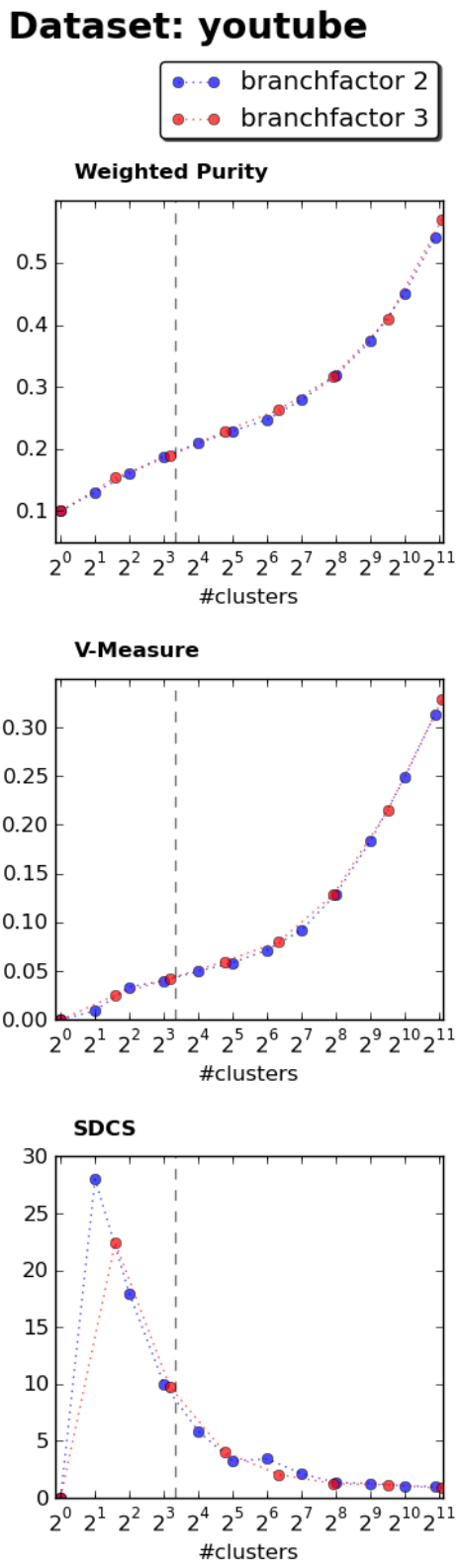
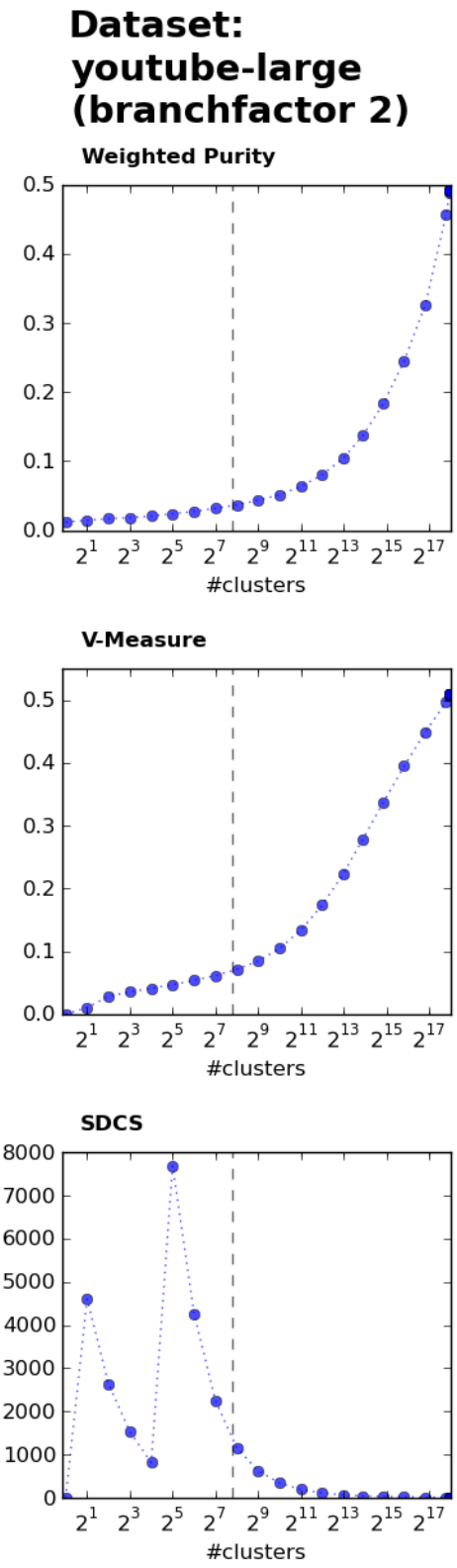


Figure 31: Results of Experiment 4 comparing different branch factors for the netclean and the caltech-256 dataset. Even though there seem to be large differences in SDCS for the caltech-256 dataset, these are more or less imperceptible, because they are still rather small compared to the large size of the dataset and its clusters.



(a) Results of Experiment 4 comparing different branchfactors for the youtube dataset.



(b) Results of the large-scale experiment using the youtube-large dataset. The proposed `hbo-k-means` algorithm meets the scalability requirement by clustering 716k images.

5 CONCLUSION

The advent of huge image and video databases has created the need for managing tools, which help the user with searching and organizing the digital data automatically by their content. One solution is browsing environments, which structure the data by image clustering. For this specific application the employed clustering algorithm needs to be highly scalable and capable of producing balanced structures to be useful for the browsing task.

One main contribution of this thesis is a comprehensive suitability study of `k-means`-based and hierarchical agglomerative clustering techniques for the described task. A second contribution is a simple but efficient approach to enforce highly balanced structures without impairing cluster quality. This approach was integrated in a hierarchical frequency sensitive variant of the online `k-means` algorithm, which we call `hbo-k-means`. This method was demonstrated to be very scalable and will also be used in a practically employed system within the FIVES Project.

5.1 DISCUSSION OF RESULTS

It has been shown that `k-means`-based methods are suitable for content-based image clustering, especially when balancing or scalability requirements are imposed as it is the case for the application of image browsing. Actually, they perform almost surprisingly well compared to the computationally more expensive hierarchical agglomerative methods, which largely suffer from producing unbalanced structures.

Hierarchical `k-means` methods do not necessarily perform worse than hierarchical agglomerative approaches. In fact, they perform clearly superior for small numbers of clusters. However, the hierarchical agglomerative methods outperform the top-down approach on lower levels of the tree. Compared to flat `k-means`, hierarchical `k-means` can be concluded to lose about 5%-28% in terms of cluster quality (Weighted Purity and V-Measure) for the ground-truth number of clusters. Thus, it can be inferred that it is not the nature of the `k-means` algorithm but the constraints of the top-down approach that are responsible for the loss of performance.

But from these constraints also the advantages of the proposed approach arise. First, the top-down approach proves to be substantially more scalable. This has been tested on a dataset comprising over 700.000 images. Second, using the proposed balancing method `bo-k-means` or its hierarchical extension `hbo-k-means` we obtain highly balanced clusters while preserving or even improving cluster quality. In addition, the strong regularization effect of balancing renders the clustering result

almost initialization-independent, significantly alleviating the inherent problem of initialization-dependence of `k-means`-based methods. Since the algorithm does not need to be repeated multiple times, this results in an even more efficient approach to image clustering.

Frequency Sensitive Competitive Learning (FSCL) methods, a broad class of existing balancing approaches, which essentially perform cluster deformation, were observed to behave unstable in very high-dimensional feature spaces often leading to degenerate solutions. In contrast, our proposed method `hbo-k-means` stably produces balanced clusters in a simple and efficient way by incorporating a cluster size sensitive additive bias. This proposed balancing approach was found to outperform FSCL methods in terms of stability.

Finally, another advantage of the proposed approach is the ability to produce a nested series of partitions of any desired branch factor. It can be observed that they have practically no effect on neither cluster quality nor balancing. Hence, a suitable branch factor can be chosen to meet particular applications needs.

5.2 FUTURE WORK

There are several open issues related to content-based image clustering that need further study. In the following paragraphs an outline of the immediate future work is presented.

First of all, it needs to be mentioned that our proposed `k-means`-based top-down approach can be massively parallelized to utilize today's multi-core machines or graphics processing units (GPU). Since top-down approaches constrain images to remain in "their" branch of the constructed clustering tree, all branches are independent. Therefore, all further clustering processes can easily be executed in parallel.

Another important issue is to effectively deal with incremental insertions and deletions in the image database. These changing and usually growing image collections are very common in practice. For large collections, recomputing the whole nested clustering is too expensive and not mandatory. For instance, it might suffice to delete specific nodes of the tree structure or fuse new images with its most similar one in the database to a new node (see [11]). Also, our online approach, where we handle one image at a time, can be seen as a first step towards evolving databases. However, we still lack a thorough investigation on how this behaves over time and how different clustering techniques might affect this behavior.

We described the complexity of image clustering due to the subjectivity of the task (recall the example of the elephant, the tuna fish and the whale). The question is, which partitions of the image collection correspond best to the users task and will ultimately satisfy him the most. This is a major problem, which also complicates the evaluation of clustering results. In this thesis, this issue was addressed by using ground-truth partitioning (on a single level) of the used datasets. However, to get a

better understanding of the subjectivity of content-based image browsing, studies are required to examine user satisfaction for this task. In addition, it should be investigated how clustering could be dynamically adapted in an efficient way to meet different users needs.

A second major problem, referred to as semantic gap, describes the "lack of coincidence between the information that one can extract from the data and the interpretation of the same data for a user in a given situation" (see [51]). It is understood that this semantic gap cannot be bridged entirely, because a comprehensive user context model does not exist for every situation. Images are usually associated with context information, such as textual captions, surrounding text (e.g. on web pages), image file names or existing folder structures, which could be used to be able to group semantically related images, which would not be deemed similar if solely relying on visual information (for clustering of "paired data" see [7]). For example, *Google Image Swirl* utilizes metadata in addition to multiple image content features to organize image search results by hierarchical clustering (see [35]). The incorporation of metadata in our system would not require major changes. Only a compound distance measure to measure the dissimilarity of content-based image features and available metadata would be required.

BIBLIOGRAPHY

[1] Ashish Agrawal and Harish Karnick. Unsupervised image clustering, 2009.

[2] Stanley C. Ahalt, Ashok K. Krishnamurthy, Prakoon Chen, and Douglas E. Melton. Competitive learning algorithms for vector quantization. *Neural Networks*, 3(3):277–290, 1990. ISSN 0893-6080.

[3] Arindam Banerjee and Joydeep Ghosh. On scaling up balanced clustering algorithms. In *In Proceedings of the SIAM International Conference on Data Mining*, pages 333–349, 2002.

[4] Arindam Banerjee and Joydeep Ghosh. Frequency-sensitive competitive learning for scalable balanced clustering on high-dimensional hyperspheres. *IEEE Transactions on Neural Networks*, 15(3):702–719, 2004.

[5] Arindam Banerjee and Joydeep Ghosh. Scalable clustering algorithms with balancing constraints. *Data Mining Knowledge Discovery*, 13(3):365–395, 2006. ISSN 1384-5810.

[6] Thomas Bärecke, Ewa Kijak, Andreas Nürnberger, and Marcin Detyniecki. VideoSOM: A SOM-based interface for video browsing. In Hari Sundaram, Milind Naphade, John R. Smith, and Yong Rui, editors, *Proc. of the 5th International Conference on Image and Video Retrieval (CIVR)*, volume 4071 of *Lecture Notes in Computer Science, LNCS*, pages 506–509. Springer, Tempe, AZ, USA, 2006.

[7] Matthew B. Blaschko and Christoph H. Lampert. Correlational spectral clustering. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 0:1–8, 2008. doi: http://doi.ieeecomputersociety.org/10.1109/CVPR.2008.4587353.

[8] Damian Borth, Christian Schulze, Adrian Ulges, and Thomas M. Breuel. Navidgator - similarity based browsing for image and video databases. In *KI '08: Proceedings of the 31st annual German conference on Advances in Artificial Intelligence*, pages 22–29, Berlin, Heidelberg, 2008. Springer-Verlag. ISBN 978-3-540-85844-7.

[9] P. S. Bradley, K. P. Bennett, and A. Demiriz. Constrained k-means clustering. Technical report, MSR-TR-2000-65, Microsoft Research, 2000.

[10] R. Calinski and J. Harabasz. A dendrite method for cluster analysis. *Communications in Statistics*, 3:1–27, 1974.

[11] Jau-Yuen Chen, Charles A. Bouman, and John C. Dalton. Hierarchical browsing and search of large image databases. *IEEE Transactions on Image Processing*, 9: 442–455, 2000.

[12] Jau-Yuen Chen, Charles A. Bouman, and Edward J. Delp. Vibe: A compressed video database structured for active browsing and search. In *IEEE Transactions on Multimedia*, pages 4–7, 2004.

[13] Yixin Chen, James Z. Wang, and Robert Krovetz. Content-based image retrieval by clustering, 2003.

[14] Ondrej Chum and Jiri Matas. Web scale image clustering – large scale discovery of spatially related images, 2008.

[15] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. Wiley Series in Telecommunications. John Wiley & Sons, New York, NY, USA, 2 edition, 1991.

[16] David L. Davies and Donald W. Bouldin. A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1(2):224–227, January 1979.

[17] M.J.L. De Hoon, S. Imoto, J. Nolan, and S. Miyano. Open source clustering software. *Bioinformatics*, 20(9):1453–1454, 2004.

[18] Ork de Rooij, Cees G. M. Snoek, and Marcel Worring. Mediamill: semantic video search using the rotorbrowser. In *CIVR '07: Proceedings of the 6th ACM international conference on Image and video retrieval*, pages 649–649, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-733-9. doi: http://doi.acm.org/10.1145/1282280.1282376.

[19] Thomas Deselaers. Features for image retrieval. Master's thesis, RWTH Aachen University, 2003.

[20] Thomas Deselaers, Daniel Keysers, and Hermann Ney. Features for image retrieval: An experimental comparison. *Information Retrieval*, 11:77–107, 2008.

[21] J. C. Dunn. Well separated clusters and optimal fuzzy-partitions. *Journal of Cybernetics*, 4:95–104, 1974.

[22] EConsultancy. 20+ mind-blowing social media statistics revisited. `http://econsultancy.com/blog/5324-20+-mind-blowing-social-media-statistics-revisited`, January 2010.

[23] C. Faloutsos, W. Equitz, M. Flickner, W. Niblack, D. Petkovic, and R. Barber. Efficient and effective querying by image content. *Journal of Intelligent Information Systems*, 3:231–262, 1994.

[24] X. Feng. Content-based image retrieval based on j. huang's color-spatial image indexing and applications, 2002.

[25] Aristides S. Galanopoulos, Randolph M. Moses, and Stanley C. Ahalt. Diffusion approximation of frequency sensitive competitive learning. *IEEE Transaction on Neural Networks*, 8(5):1026–1030, 1997.

[26] John F. Gantz, C. Chute, A. Manfrediz, S. Minton, D. Reinsel, W. Schlichting, and Anna Toncheva. The diverse and exploding digital universe, March 2008. URL `http://www.emc.com/collateral/analyst-reports/diverse-exploding-digital-universe.pdf`.

[27] Jacob Goldberger, Hayit Greenspan, and Shiri Gordon. Unsupervised image clustering using the information bottleneck method. In *DAGM-Symposium*, pages 158–165, 2002.

[28] Jacob Goldberger, Shiri Gordon, and Hayit Greenspan. Unsupervised image-set clustering using an information theoretic framework. *IEEE Transactions on Image Processing*, 15(2):449–458, 2006.

[29] G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. Technical Report 7694, California Institute of Technology, 2007. URL `http://authors.library.caltech.edu/7694`.

[30] Johann Hofmann and Moataz Ali. An extensive approach to content based image retrieval using low- & high-level descriptors. Master's thesis, IT University of Göteborg, 2006.

[31] Jing Huang, S. Ravi Kumar, Mandar Mitra, Wei-Jing Zhu, and Ramin Zabih. Image indexing using color correlograms. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 0:762–768, 1997.

[32] Jing Huang, S. Ravi Kumar, and Ramin Zabih. Automatic hierarchical color image classification. *EURASIP J. Appl. Signal Process.*, 2003:151–159, 2003.

[33] Alexa Internet. Alexa traffic rank for youtube (three month average). `http://www.website-monitoring.com/blog/2010/05/17/youtube-facts-and-figures-history-statistics/`, August 2010. Retrieved August 30, 2010.

[34] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: A review, 1999.

[35] Yushi Jing, Henry Rowley, Charles Rosenberg, Michele Covell Jingbin Wang, Liu Yi, and Marius Pasca. Google image swirl, a large-scale content-based image browsing engine. Demo at IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2010.

[36] Thomas Käster, Volker Wendt, and Gerhard Sagerer. Comparing clustering methods for database categorization in image retrieval. In Bernd Michaelis and Gerald Krell, editors, *DAGM-Symposium*, volume 2781 of *Lecture Notes in Computer Science*, pages 228–235. Springer, 2003. ISBN 3-540-40861-4.

[37] Michael Kearns, Yishay Mansour, and Andrew Y. Ng. An information-theoretic analysis of hard and soft assignment methods for clustering. In *In Proceedings of the Thirteenth Annual Conference on Uncertainty in Artificial Intelligence*, pages 282–293. Morgan Kaufmann, 1997.

[38] Donald E. Knuth. Computer Programming as an Art. *Communications of the ACM*, 17(12):667–673, December 1974.

[39] Ferenc Kovács, Csaba Legány, and Attila Babos. Cluster validity measurement techniques, 2005.

[40] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–110, 2004.

[41] B.S. Manjunath, J.-R. Ohm, V.V. Vasudevan, and A. Yamada. Color and texture descriptors. *Circuits and Systems for Video Technology, IEEE Transactions on*, 11(6): 703 –715, jun 2001.

[42] U. Maulik and S. Bandyopadhyay. Performance evaluation of some clustering algorithms and validity indices. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(12):1650–1654, 2002.

[43] Henning Müller, Stephane Marchand-Maillet, and Thierry Pun. The truth about corel - evaluation in image retrieval. In Michael Lew, Nicu Sebe, and John Eakins, editors, *Image and Video Retrieval*, volume 2383 of *Lecture Notes in Computer Science*, pages 38–49. Springer Berlin / Heidelberg, 2002.

[44] Website Monitoring. Youtube facts & figures (history & statistics). `http://www.website-monitoring.com/blog/2010/05/17/youtube-facts-and-figures-history-statistics/`, May 2010.

[45] David Nister and Henrik Stewenius. Scalable recognition with a vocabulary tree. In *CVPR '06: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2161–2168, Washington, DC, USA, 2006. IEEE Computer Society. ISBN 0-7695-2597-0. doi: http://dx.doi.org/10.1109/CVPR.2006.264.

[46] Andrew Rosenberg and Julia Hirschberg. V-measure: A conditional entropy-based external cluster evaluation measure, 2007.

[47] P. Scheunders and S. De Backer. High-dimensional clustering using frequency sensitive competitive learning. *Pattern Recognition*, 32:193–202, 1999.

[48] S .Z. Selim and M. A. Ismail. K-means-type algorithms: A generalized convergence theorem and characterization of local optimalityy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:81–87, 1984.

[49] T. Sikora. The mpeg-7 visual standard for content description-an overview. *Circuits and Systems for Video Technology, IEEE Transactions on*, 11(6):696–702, 2001.

[50] J. Sivic, B. C. Russell, A. A. Efros, A. Zisserman, and W. T. Freeman. Discovering object categories in image collections. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2005.

[51] Arnold W. M. Smeulders, Marcel Worring, Simone Santini, Amarnath Gupta, and Ramesh Jain. Content-based image retrieval at the end of the early years. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:1349–1380, 2000.

[52] M. A. Stricker and M. Orengo. Similarity of color images. In W. Niblack and R. C. Jain, editors, *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, volume 2420 of *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, pages 381–392, March 1995.

[53] Michael J. Swain and Dana H. Ballard. Color indexing. *International Journal of Computer Vision*, 7:11–32, 1991.

[54] Hideyuki Tamura, Shunji Mori, and Takashi Yamawaki. Textural features corresponding to visual perception. *Systems, Man and Cybernetics, IEEE Transactions on*, 8(6):460 –473, June 1978.

[55] Luc Vincent. Taking online maps down to street level. *Computer*, 40(12):118–120, 2007. ISSN 0018-9162. doi: http://doi.ieeecomputersociety.org/10.1109/MC.2007.442.

[56] Minerva M. Yeung, Boon-Lock Yeo, and Bede Liu. Extracting story units from long programs for video browsing and navigation. In *ICMCS '96: Proceedings of the 1996 International Conference on Multimedia Computing and Systems*, pages 296–305, 1996.

DECLARATION

I declare that this document has been composed by myself, and describes my own work, unless otherwise acknowledged in the text. It has not been accepted in any previous application for a degree. All verbatim extracts have been distinguished by quotation marks, and all sources of information have been specifically acknowledged.

Kaiserslautern, September 2, 2010

Christopher Tim Althoff