chist-era

Multimedia Fusion

Mihai Lupu*, Ralf Bierig*, Pinar Duygulu†, Adrian Popescu‡

*Vienna University of Technology, Austria

†Bilkent University, Turkey

‡CEA, LIST, LVIC France

Contacts: lupu@ifs.tuwien.ac.at, duygulu@cs.bilkent.edu.tr, adrian.popescu@cea.fr

MUCKE Project, Deliverable 2.3

07/07/2014

Contents

Abstract

Multimodal data is easily integratable by the human mind, but very difficult to do by the computer. The assumption is that this facility of the mind resides in its ability to map everything to the same space of concepts: reading the word flower automatically and involutanrily bring the image, or even the perfume of a flower in the mind; seeing a flower prepares the neural path for accessing the written term or even reminds us a piece of text, a poem, mentioning flowers. Therefore, in MUCKE, the efforts in integrating multiple modalities of information revolve around mapping data into the same space. This deliverable summarizes these efforts and some of their initial results.

1 INTRODUCTION

Multimodal information access is desireable because it is a natural way for human to interact with the world - we use all our senses at once, rather than sequentially. Computers are not generally able to do that with any degree of utility (with perhaps the experience of video processing, where sound and image processing have been used to enrich each other).

While different modalities of information (in particular text and images) often occur together in the same document (scientific paper, patent, blog, etc.), search through these modalities is usually done for each modality in isolation. It is well known that combining information from multiple modalities can assist in retrieval tasks, e.g. results of the ImageCLEF campaign's photographic retrieval task have shown that combining image and text information results in better retrieval than text alone [37]. There are two basic approaches to fusing information from multiple modalities: early fusion and late fusion [8]. For early fusion, modalities are mixed at feature level, by for example concatenating the feature vectors from two or more modalities [19]. For late fusion, different search systems operate on the different modalities, and the output of the systems are fused. In general, the approaches used to combine these modalities for retrieval are almost always ad-hoc.

Late fusion is more widely used, as it avoids the difficulties of working directly in a single fused feature space. Instead, it fuses the output of retrieval systems working on the different modalities separately. This includes reordering results based on information from a second modality, and various arithmetic combinations of scores obtained by the systems treating the different modalities. Clinchant et al. [6] propose and test a number of late fusion approaches involving the sum or product combination of weighted scores from text and image retrieval systems. A difficulty with the arithmetic combination are the parameters in the form of the weights for different modalities, where the weights must be fixed in advance. Fixed parameter values have the disadvantage that different queries may be best answered by different modality weight combinations. These weights are also sensitive to the performance of the retrieval systems used for the various modalities [8]. Any late fusion approach would have to deal with some of the problems that distributed IR had to deal with. In particular, identifying the optimal way to merge lists of ranked documents (or parts thereof) arriving from different search engines. Having calculated scores based on the images or the text in the documents, merging the lists to obtain a final set of relevant documents requires a mapping between these scores and the probability of relevance [33].

A disadvantage of late fusion is that a separate query is needed for each modality, so that for example to find a picture of a cat in a database of annotated images, one would need to provide a picture of a cat and text about the cat. There are ways of getting around this limitation, such as choosing the images for the top returned text documents as seeds in an image search [8], but these are generally ad-hoc. With early fusion, a query would not have to contain elements from all modalities in the dataset. To continue the previous example, pictures of a cat could be found only with text input. However, individual queries for each modality may be desireable in some domains (e.g. particularly important for professional searchers, who have a high degree of sophistication in

the creation of their queries and who need to prove that all information was properly searched).

Early fusion suffers from the problem that text encoded using the bag of words technique tends to sparsely inhabit a large feature space, while features from the non-text modalities tend to have a denser distribution in a small feature space. It is however possible to represent images sparsely in higher-dimensional feature spaces through the use of bags of "visual words" [7], where the visual words are obtained by clustering local features extracted from the images. The simplest approach to early fusion is to simply concatenate the feature vectors from different modalities. However, concatenated feature vectors become less distinctive, due to the curse of dimensionality [8], making this approach rather ineffective. A solution proposed in [30] is to transform the feature vectors to reduce the dimension of the text feature vectors and increase the dimension of the image feature vectors using the minimum description length (MDL) principle. Hwang and Grauman [15] use kernel canonical correlation analysis (CCA) to create a "semantic space" in which tags and corresponding image features lie close to each other. These papers however always have a small amount of text associated with each image, in the form of image tags or six seconds of spoken text around a video keyframe. Rasiwasia et al. [39] use canonical correlation analysis to fuse image and text features. While they use relatively lengthy paragraphs of text associated with each image, the experimental dataset consists of only 2866 documents (image and associated paragraph) classified into 10 subject categories. The dimensionality of the feature spaces is small, as LDA was used to reduce the text dimensionality to 10 (given the a priori information on the number of categories), and the visual word vocabulary was 128. The tensor product (inspired by quantum IR approach [45]) was used in [47], leading to feature space with dimensionality equal to the product of the dimensionalities of the image and text feature spaces. For the experiments, a dataset of 20,000 annotated images was used, but with low-dimensional colour visual features and various rules to reduce the dimensionality of the text features. It is clear from the relatively small-scale experiments in the literature that scaling up early fusion to large amounts of data and high-dimensional feature spaces is a challenge, and many of the solutions adopted to reduce feature vector dimensionality are based on a priori knowledge about the dataset used.

Addressing the issues of early or late fusion is done in MUCKE via the MUCKE framework which is a multimedia retrieval system framework incorporating components for processing multimedia content in different modalities and languages. The framework provides concept-based information retrieval facilities that applies credibility information for result reranking. The architecture combines both a direct user interface and a batched evaluation interface for reproducible research in multimedia IR [1].

Fusion of different modalities will ultimately come back to mapping them to a common semantic space and therefore the primary question is what this space may be and what are concepts in it? Looking at the image processing and retrieval community, we observe that by "extracting concepts" one often means extracting terms, while in the text processing and retrieval community "extracting concepts" generally means mapping to a set of abstract unique notations, either statistically generated (based on distributional semantics [16]) or manually or semi-automatically generated (often a

subset of Linked Open Data[1], such as DBpedia or Cyc URIs). MUCKE follows multiple approaches, covering both of these options.

The remainder of this report is structured as follows. First, in Section 2 we describe efforts to map text into a semantic space, by distributional semantics and deep learning, including the options for using the extracted features in a learning to rank approch. Then, in Section 3 we focus on mapping images to a semantic space. Finally, Section 4 provides some considerations about conceptual indexing, which will be then developped in the upcoming Deliverables on the topic.

2 TEXT IN CONCEPT SPACE

This section of the report starts from the text to map the data into a semantic information space. In such a context, the noise in the data, and the uncertainty of the relevance scores it generates, comes from synonymy and polysemy and the effort is to generate a unique identifier for each concept.

2.1 DISTRIBUTIONAL SEMANTICS

There is something fundamentally attractive to the concept of statistical semantics, to the idea of computers simulating human behaviour, or, in this case, human understanding. The original LSI came first as a mathematical best-approximation of the original term-document matrix, and was only later analysed as a functional model for cognitive activities [21]. Conversely, Random Indexing starts from philosophical ideas of meaning. In their 2001 article, Karlgren and Sahlgren [17] start from Ludwig Wittgenstein's 1953 *Philosophical Investigations* and define knowing the meaning of a word as knowing how to use it correctly in context. Note that the definition refers to the use of meanings, rather than the meaning of meaning.

Random Indexing works essentially in two steps: first, an N-dimensional random label is assigned to each word type in the data. This contains only a small random number k of +1 or -1 values among many 0s. Second, for every term occurrence, a context vector is created by summing together the vectors of the terms in its context. This way (and as we will see, with potential repetition of the two steps) we obtain a vector considered to represent the context of the term, and therefore, its meaning.

All statistical semantics methods (and we include here the works of Schütze [42] and Lund [27]) are therefore based on the philosophical wisdom that the meaning of words can be represented by the sum of contexts in which they appear. The difference between all of them is given by the interpretation and implementation of the "sum" and "context".

Such methods have been shown to display functional behaviour comparable to users speaking English as a foreign language [21, 17]. The next step is to see how well these methods perform in a search environment that requires expert users. We have low expectations regarding the quality of

[1] `http://linkeddata.org/`

the results, but at the same time, such a study needs to be done, and it is to some extent surprising that we could not find something in the existing literature.

We have tested this approach on a technically difficult collection of patents [28]. In this sense, we have observed that applying the distributional semantics approach dirrectly to document similarity is not resulting in results comparable with the state of the art. However, random indexing is particularly suited at identifying conceptually similar terms. The statistical semantics are able to first, identify variants of the same term and second, distinguish different meanings for different variants of the same term. We show an example for each of the two cases, to illustrate what we mean. For the first point, Listing 1 shows the 10 most similar terms to the term `coatings`.

Listing 1: Top 10 similar terms to "coatings"

```
1.0:coatings
0.9999339:rubs
0.9999338:coating
0.9999328:acrylics
0.9999271:vinyls
0.9999268:cratering
0.9999251:distinctness
0.9999246:blistering
0.9999235:pompano
0.9999234:cyanamid
```

As can be seen, the singular form appears very high in the list, surrounded by different types of coatings.

Regarding the second point, we take the example of "crystal" and "crystals". The first is generally used in optics, while the second in chemistry. Listings 2 and 3 show the two top-10 similar word lists. Note how Listing 2 identifies optical devices (and misspellings at positions 2 and 3) , while Listing 3 refers mostly to the process of crystallisation (and it even manages to identify a poorly tokenized version of the term, at position 5 and 10).

Listing 2: Top 10 similar terms to "crystal"

```
1.0:crystal
0.9999378:cyrstal
0.9999305:crytal
0.9999022:nicol// a type of prism
0.9999014:jjap
0.9999006:nicols
0.9998996:nematic// a type of liquid crystal
0.9998943:uniaxial//minerals that form crystals used in optics
0.9998894:cb15//a particular liquid crystal
```

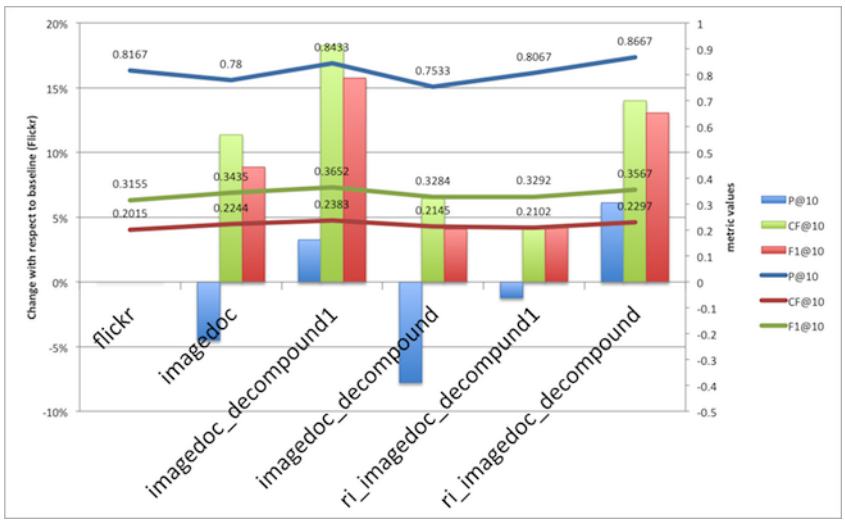


FIGURE 1: Initial results on the MediaEval Retrieving Diverse Images task

```
0.9998887: anisotropy
```

Listing 3: Top 10 similar terms to "crystals"

```
1.0: crystals
0.9998632: supersaturation
0.9998519: crystallizing
0.9998281: supersaturated
0.9998213: crys
0.9998193: purer
0.9998166: soda
0.9998120: crystallize
0.9998105: crystallizers
0.9998081: tals
```

Recently, Zadeth [50] has also observed that distributional semantics are particularly suited at classifying terms. More generally, we hypothesize that the lack of solid results in the document similarity case is a result of the size of the documents in the patent collection. Therefore, we continued and tested the approach on the image annotation set of the MediaEval retrieving diverse images task. Figure 1 shows these initial results. Random indexing provided the greatest improvement compared to the baseline (the original Flickr ranking) and comparable to the other methods presented previously in the evaluation campaign.

2.2 DEEP LEARNING

Word2Vec further expands the latent semantics approach while being highly incremental and scalable [31]. When trained on large datasets, it is also possible to capture many linguistic subtleties (e.g. relations between cities to their counties) that allow basic arithmetic operations within the

model. This, in principle, allows exploiting the implicit knowledge within corpora. All of these methods represent the words in a vector spaces.

This sets these methods apart from approaches that use external knowledge, such as WordNet or OpenCyc, for determining word meanings by explicitly expressed, human-entered knowledge.

2.3 LEARNING TO RANK

Learning to rank refers to machine learning techniques for training a model in a ranking task. Due to importance of ranking problems, learning to rank has been drawing broad attention in the machine learning community recently.

In the learning to rank approach, the ranking problem is transformed to classification, regression and ordinal classification, and existing methods and techniques for solving machine learning problems are applied. As Hang [13] points out, the relation between learning to rank and ordinal classification is that, in ranking, one cares more about accurate ordering of objects, while in ordinal classification, one cares more about accurate ordered-categorization of objects.

The first step in accumulating data required for learning to rank, is relevance judgments, normally done by human annotators. Lie [25] presents the three main strategies in learning to rank:

- *Relevance degree*: In this method, the annotator specifies whether an object is relevant or not to the query. It can be either in binary judgment or by specifying the degree of relevance (e.g., Perfect, Excellent, Good, Fair, or Bad).

- *Pairwise preference*: The annotator compares a pair of objects in order to specify which one is more relevant with regards to a query.

- *Total order*: The annotator specifies the total order of all objects with respect to a query by rating each object.

Among the three mentioned kinds of judgments, the first one is the most popularly used judgment since is the easiest to obtain, while the third one is more accurate but laborious for human annotators. In our case, we have used the total order method because our ranked lists consisted of only 3 translators.

The learning to rank techniques are categorized in three main groups: *Pointwise*, *Pairwise* and *Listwise*.

In the pointwise approach, the ranking problem is transformed to classification, regression or ordinal classification. Therefore, the group structure of ranking is ignored in this approach [13]. Here, linear or polynomial regression are widely used methods.

The pairwise approach transforms the ranking problem into pairwise classification or regression. In fact, it cares about the relative order between two documents. Similar to the pointwise approach, the pairwise method also ignores the group structure of ranking [13]. Here is a brief explanation of some pairwise algorithms:

- *RankNet* [4]: Widely applied by commercial search engines, it uses gradient descent method and neural network to model the underlying ranking function.
- *RankBoost* [12]: It adopts AdaBoost algorithm for the classification over the object pairs.
- *LambdaRank* [3]: It considers the evaluation measures to set its pair weight. In particular, the evaluation measures (which are position based) are directly used to define the gradient with respect to each document pair in the training process.
- *LambdaMART* [48]: It combines the strengths of boosted tree classification and LambdaRank.

The listwise approach takes the entire set of documents associated with a query in the training data as the input and predicts their ground truth labels [25]. In contradiction to two previous approaches, it maintains the group structure of ranking. In addition, ranking evaluation measures can be more directly incorporated into the loss functions in learning [13]. In the following, two common listwise algorithms are briefly discussed:

- *AdaRank* [49]: It applies the evaluation measures on the framework of Boosting and focuses on effectively optimization.
- *ListNet* [5]: It uses different probability distributions in order to define the loss function.

Lie [25] compares the algorithms by applying on different data-sets. It concludes that listwise techniques are in general the most effective among the others. However, the choice of the learning evaluation measure and the rank cutoff may have a noticeable impact on the effectiveness of the learned model [29].

We have tested the different learning to rank approaches on a multilingual test collection for a real-word test case [40]. Table 1 shows the results according to Normalized Discounted Cummulative Gain (NDCG) and Estimated Reciprocal Rank (ERR). At the time of writing of this deliverable, the Retrieving Diverse Relevant Images evaluation campaign is underway and the results for this corpus are not yet available.

3 IMAGES IN CONCEPT SPACE

3.1 CONCEPTMAP: MINING NOISY WEB DATA FOR CONCEPT LEARNING

The need for manually labelled data continues to be one of the most important limitations in large scale recognition. Alternatively, images are available on the Web in huge amounts. This fact recently attracted many researchers to build (semi-)automatic methods to learn from web data collected for a given concept. However, there are several challenges that makes the data collections gathered from web different from the hand crafted datasets. Images on the web are "in the wild" inheriting

TABLE 1: Results of applying Learning to Rank methods based on NDCG and ERR evaluation measures

Method	NDCG@3		ERR@3	
	Result	Random	Result	Random
Linear Regression	**0.935**	0.833	**0.451**	0.375
RankNet	0.876	0.834	0.394	0.378
RankBoost	0.909	0.831	0.432	0.374
LambdaMART	0.93	0.832	0.447	0.373
ListNet	0.915	0.831	0.439	0.375
AdaRank	0.857	0.83	0.399	0.373
Random Ranker	0.832	0.832	0.375	0.378

FIGURE 2: Example Web images for (a) spotted, (b) office, (c) motorbikes, (d) Angelina Jolie.

all types of challenges due to variations and effects. Since usually images are gathered based on the surrounding text, the collection is very noisy with several visually irrelevant images as well as images corresponding to different characteristic properties of the concept (Figure2).

For the queried data for automatic learning of concepts, we propose a novel method to obtain a representative groups with irrelevant images removed. Our intuition is that, given a concept category by a query, although the list of images returned include irrelevant ones, there will be common characteristics shared among subset of images. Our main idea is to obtain visually coherent subsets, that are possibly corresponding to semantic sub-categories, through clustering and to build models for each sub-category (see Figure3). The model for each concept category is then a collection of multiple models, each representing a different aspect.

To retain only the relevant images that describe the concept category correctly, during clustering we need to remove outliers, i.e. irrelevant ones. The outliers may resemble to each other while not being similar to the correct category resulting in a **outlier cluster**. Alternatively, outlier images could be mixed with correct category images inside **salient clusters** corresponding to relevant ones. These images, that we refer to as **outlier elements**, should also be removed for the quality data for learning.

We propose a novel method **Concept Maps (CMAP)** for which organises the data by purifying it not only from outlier clusters but also from outlier elements in salient clusters. CMAP captures category characteristics through organising the set of given instances into sub-categories pruned from irrelevant instances. It is a generic method that could be applied on any type of concept from

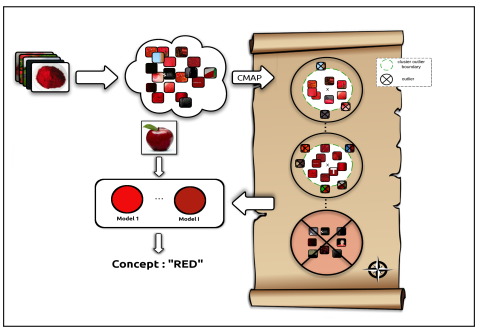


FIGURE 3: Overview of our framework for concept learning shown on example concept "Red". Concept Map (CMAP) organises the images collected from web for the given text query into clusters which are pruned from outlier elements inside salient clusters and outlier clusters. Each cluster is then used as a sub-model for learning and localising the concept in a novel image.

low-level attributes to high level object and scene categories as well as faces.

3.1.1 CONCEPT MAPS

We propose CMAP which is inspired from the well-known Self Organizing Maps (SOM) [18]. In the following, SOM will be revisited briefly, and then CMAP will be described.

Revisiting Self Organizing Maps (SOM): Intrinsic dynamics of SOM are inspired from developed animal brain where each part is known to be receptive to different sensory inputs and which has a topographically organized structure[18]. This phenomena, i.e. "receptive field" in visual neural systems [14], is simulated with SOM, where neurons are represented by weights calibrated to make neurons sensitive to different type of inputs. Elicitation of this structure is furnished by competitive learning approach.

Consider input $X = \{x_1, .., x_M\}$ with M instances. Let $N = \{n_1, ..., n_K\}$ be the locations of neuron units on the SOM map and $W = \{w_1, ..., w_K\}$ be the associated weights. The neuron whose weight vector is most similar to the input instance x_i is called as the winner and denoted by $\hat{v}$. Weights of the winner and units in the neighbourhood are adjusted towards the input at each iteration t with delta learning rule.

$$w_j^t = w_j^{t-1} + h(n_i, n_{\hat{v}} : \epsilon^t, \sigma^t)[x_i - w_j^{t-1}] \tag{1}$$

Update step is scaled by the window function $h(n_i, n_{\hat{v}} : \epsilon^t, \sigma^t)$ for each SOM unit, inversely proportional to the distance to the winner (Eq.2). Learning rate ϵ is a gradually decreasing value, resulting in larger updates at the beginning and finer updates as the algorithm evolves. σ^t defines the neighbouring effect so with the decreasing σ, neighbour update steps are getting smaller in each epoch. Note that, there are different alternatives for update and windows functions in SOM literature.

$$h(n_i, n_{\hat{v}} : \epsilon^t, \sigma^t) = \epsilon^t \exp \frac{-||n_j - n_{\hat{v}}||^2}{2\sigma^{t^2}} \tag{2}$$

Clustering and outlier detection with CMAP: We introduce excitation scores $E = \{e_1, e_2, \ldots, e_K\}$ where e_j, the score for neuron unit j, is updated as in Eq.3.

$$e_j^t = e_j^{t-1} + \rho^t(\beta_j + z_j) \tag{3}$$

As in SOM, window function is getting smaller with each iteration. z_j is the activation or win count for the unit j, for one epoch. ρ is learning solidity scalar that represents the decisiveness of learning with dynamically increasing value, assuming that later stages of the algorithm has more impact on the definition of salient SOM units. ρ is equal to the inverse of the learning rate ϵ. β_j is the total measure of the activation of jth unit in an epoch, caused by all the winners of the epoch but the neuron itself (Eq.4).

$$\beta_j = \sum_{\hat{v}=1}^{u} h(n_j, n_{\hat{v}}) z_{\hat{v}} \tag{4}$$

At the end of the iterations, normalized e_j is a quality value of a unit j. Higher value of e_j indicates that total amount of excitation of the unit j in whole learning period is high thus it is responsive to the given class of instances and it captures notable amount of data. Low excitation values indicate the contrary. CMAP is capable of detecting outlier units via a threshold θ in the range $[0, 1]$.

Let $C = \{c_1, c_2, \ldots, c_K\}$ be the cluster centres corresponding to each unit. c_j is considered to be a **salient cluster** if $e_j \geq \theta$, and an **outlier cluster** otherwise.

The excitation scores E are the measure for saliency of neuron units in CMAP. Given the data belonging to a category, we expect that data is composed of sub-categories that share common properties. For instance `red` images might include tones to be captured by clusters but they are supposed to share a common characteristics of being red. For the calculation of the excitation scores we use individual activations of the units as well as the neighbouring activations. Individual activations measure being a salient cluster corresponding to a particular sub-category, such as `lighter red`. Neighbourhood activations count the saliency in terms of the shared regularity between sub-categories. If we don't count the neighbourhood effect, some unrelated clusters would be called salient, e.g. noisy white background patches in `red` images.

Outlier instances in salient clusters (**outlier elements**) should also be detected. After the detection of outlier neurons, statistics of the distances between neuron weight w_i and its corresponding instance vectors is used as a measure of instance divergence. If the distance between the instance vector x_j and its winner's weight $\hat{w}_i$ is more than the distances of other instances having the same winner, x_j is raised as an outlier element. We exploit box plot statistics, similar to [32]. If the distance of the instance to its cluster's weight is more than the upper-quartile value, then it is an outlier. The portion of the data, covered by the upper whisker is decided by τ.

CMAP provides good basis of cleansing of poor instances whereas computing cost is relatively smaller since an additional iteration after clustering phase is not required. All the necessary information (excitation scores, box plot statistics) for outliers is calculated at runtime of learning. Hence, CMAP is suitable for large scale problems.

CMAP is also able to estimate number of intrinsic clusters of the data. We use PCA as a simple heuristic for that purpose, with defined variance ν to be retained by the selected first principle components. Given data, principle components describing the data with variance ν is used as the number of clusters for the further processing of CMAP. If we increase ν, CMAP latches more clusters.

$$Num.Clusters = \max_{q} \left(\frac{\sum_{i=1}^{q} \lambda_i}{\sum_{j=1}^{p} \lambda_j} \leq \nu \right) \tag{5}$$

q is the number of top principle components selected after PCA and p is the dimension of instance vectors. λ is the eigenvalue of corresponding component.

Algorithm 1: CMAP

In the real code we use vectorized implementation whereas we write down iterative pseudo-code for the favour of simplicity.
Input: X, θ, τ, K, T, ν, σ^{init}, ϵ^{init}
Output: $OutlierUnits, Mapping, W$
set each item z_i in Z to 0
$u \leftarrow estimateUnitNumber(X, variation)$
$W \leftarrow randomInit(u)$
while $t \leq T$ **do**
  $\epsilon^t \leftarrow computeLearningRate(t, \epsilon^{init})$
  $\rho^t \leftarrow 1/\epsilon^t$
  set each item β_i in B to 0
  select a batch set $X^t \subset X$ with K instances
  for *each* $x_i \in X$ **do**
    $\hat{w}_i^t \leftarrow findWinner(x_i, W)$
    $\hat{v} \leftarrow min_j(||x_i - w_j||)$
    increase win count $z_{\hat{w}^t} \leftarrow z_{\hat{w}_i^t} + 1$
    increase win count $z_{\hat{v}} \leftarrow z_{\hat{v}} + 1$
    for *each* $w_k \in W$ **do**
      $\beta_k^t = \beta_k^t + h(n_k, n_{\hat{v}})$
      $w_k = w_k + h(n_k, n_{\hat{v}})||x_i - w_{\hat{v}}||$
    end
  end
  for *each* $w_j \in W$ **do**
    $e_j^t = e_j^{t-1} + \rho^t(\beta_j^t + z_j)$
  end
  $t \leftarrow t + 1$
end
$W_{out} \leftarrow thresholding(E, \theta)$
$W_{in} \leftarrow W \setminus W_{out}$
$Mapping \leftarrow findMapping(W_{in}, X)$
$Whiskers \leftarrow findUpperWhiskers(W_{in}, X)$
$X_{out} \leftarrow findOutlierIns(X, W_{in}, Whiskers, \tau)$
return $W_{out}, X_{out}, Mapping, W$

3.1.2 CONCEPT LEARNING WITH CMAP

We utilise the clusters, that are obtained through CMAP as presented above, for learning sub-models in concepts. We exploit the proposed framework for learning of attributes, scenes, objects and faces. Each task requires the collection of data, clustering and outlier detection with CMAP, and training of sub-models from the resulting clusters. In the following, first we will describe the attribute learning, and then describe the differences in learning other concepts.

Learning low-level attributes: Most of the methods require learning of visual attributes from labelled data, and cannot eliminate human effort. Here, we describe our method in learning attributes from web data without any supervision.

We collect web images through querying colour and texture names. The data is weakly labelled,

with the labels given by queries. Hence, there are irrelevant images in the collection, as well as images with a tiny portion corresponding to the query keyword.

Each image is densely divided into non-overlapping fixed-size patches to sufficiently capture the required information. We assume that the large volume of the data itself is sufficient to provide instances at various scales and illuminations, thus we did not perform any scaling or normalisation. The collection of patches extracted from all images for a single attribute is then given to CMAP to obtain clusters which are likely to capture different characteristics of the attribute as removing the irrelevant ones.

Each cluster obtained through CMAP is used to train a separate classifier. Positive examples are selected as the members of the cluster and negative instances are selected among the outliers removed by CMAP and also elements from other categories.

Learning scene categories: To show CMAP capability on higher level concepts, we target scene categories. In this case, we use the entire images as instances, and aim to discover groups of images each representing a different property of the scene, at the same time by eliminating the images that are either spurious. These clusters are then used as models similar to the attribute learning.

Learning object categories: In the case of objects, we detect salient regions on each image via [9], to eliminate background noise. Then these salient regions are fed into CMAP framework for clustering.

Learning faces: We address the problem of learning faces associated with a name -which is generally referred to face naming in the literature-, through finding salient clusters in the set of images collected from web through querying the name. Here, the clusters are likely to correspond to different poses and possibly different hair and make-up style differences as well as ageing effects. Note that this task is not the detection of faces, but recognition of faces for a given name. We detect the faces in the images, and only use a single face with the highest confidence for each image.

3.1.3 QUALITATIVE EVALUATION OF CLUSTERS

As Figure4 depicts, CMAP captures different characteristics of concepts in separate salient clusters, while eliminating outlier clusters that group irrelevant images coherent among themselves, as well as outlier elements wrongly mixed with the elements of salient clusters . CMAP detects different shades of "Brown" and eliminates some superiors elements belonging the different colors. For the "Vegetation" and "Bedroom", CMAP again divides the visuals elements with respect to structural and angular properties. Especially for "bedroom", each cluster is able to capture different view-angle of the images as it successfully removes outlier instances with some of little mistakes that are belonging to the label but not representative (circular bed in very shiny room) for the concept part. On more difficult tasks of grouping objects and faces, CMAP is again successful in eliminating outlier elements and outlier clusters as shown in Figure5.

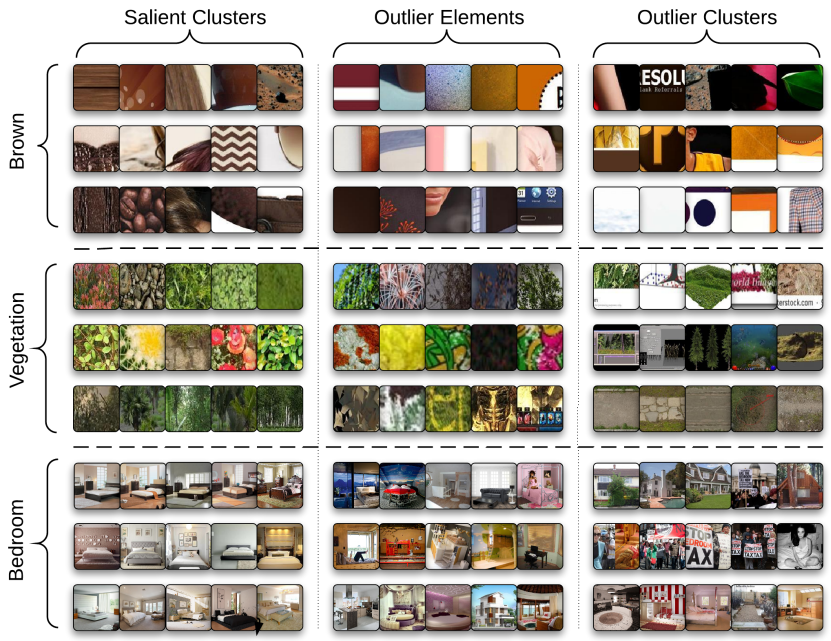


FIGURE 4: For colour and texture attributes *brown* and *vegetation* and scene concept *bedroom*, randomly sampled images detected as (i) elements of **salient clusters**, (ii) elements of **outlier clusters**, and (iii) **outlier elements** in salient clusters.

3.1.4 ATTRIBUTE LEARNING

Datasets and representation: For a better comparison, we collected images from Google for 11 distinct colours as in [44] and 13 textures. We included the terms "colour" and "texture" in the queries, such as "red colour", or "wooden texture". For each attribute, 500 images are collected. In total we have 12000 web images. Each image is divided into 100x100 non-overlapping patches. Unlike [44], we didn't apply gamma correction. For colour concepts we use 10x20x20 bins Lab colour histograms and for texture concepts we use BoW representation for densely sampled SIFT [26] features with 4000 words. We keep the feature dimensions high to utilise from the over-complete representations of the instances with L1 norm linear SVM classifier.

Attribute recognition on novel images: The goal of this task is to label a given image with a single attribute name. Although there may be multiple attributes in a single image, for being able to compare our results on benchmark data-sets we consider one attribute label per image. For this purpose, first we divide the test images into grids in three levels using spatial pyramiding [22]. Non-overlapping patches (with the same size of training patches) are extracted from each grid of all three levels. Recall that, we have multiple classifiers for each attribute trained on different salient clusters. We run all the classifiers on each grid for all patches. Then, we have a vector of confidence values for each patch, corresponding to each particular cluster classifier. We sum those confidence vectors of each patch in the same grid. Each grid at each level is labelled by the maximum confidence classifier among all the outputs for the patches. All of those confidence values are then merged with

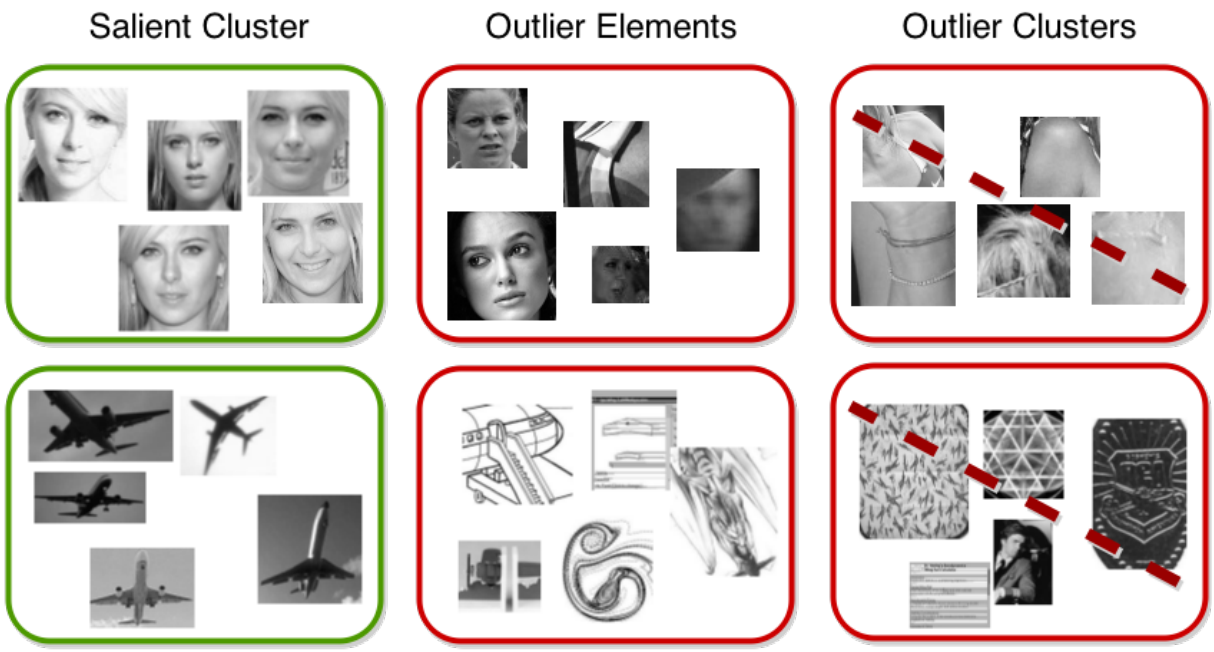


FIGURE 5: CMAP results for object and face examples. Left columns shows one example of salient cluster. Middle column shows outlier instances captured from salient clusters. Right column is the detected outlier clusters.

a weighted sum to a label for the entire image. $D^i = \sum_{l=1}^{3} \sum_{n=1}^{N_l} \frac{1}{2^{3-l}} h_i e^{-(\hat{x}-x)/2\sigma^2}$ Here, N_l is the grid number for level l and h_i is the confidence value for grid i. We include a Gaussian filter, where $\hat{x}$ is center of the image and x is location of the spatial pyramid grid, to give more priority to the detections around the center of the image for reducing noisy background effect.

For comparison with previous work, we use three different datasets. The first dataset is Bing Search Images curated by ourselves from the top 35 images returned with the same queries we used for initial images. This set includes 840 images in total for testing. Second dataset is Google Colour Images [44] previously used by [44] for learning colour attributes. Google Colour Images includes 100 images for each color name. We used the whole data-sets only for testing of our models learned on a possibly different set that we collected from Google, contrary to [44]. The last dataset is sample annotated images from ImageNet [41] for 25 attributes. To test the results on a human labelled dataset, we use Ebay dataset provided by [44] which has labels for the pixels in cropped regions. It includes 40 images for each colour name.

Figure 6 compares the overall accuracy of the proposed method (**CMAP**) with three other methods on the task of attribute learning. As the baseline (**BL**), we use all the images returned for the concept query to train a single model. As expected, the performance is very low suggesting that a single model trained by crude noisy web images performs poorly and the data should be organised to train at least some qualified models from coherent clusters in which representative images are grouped. As two other methods for clustering the data, we used k-means (**KM**) and original SOM

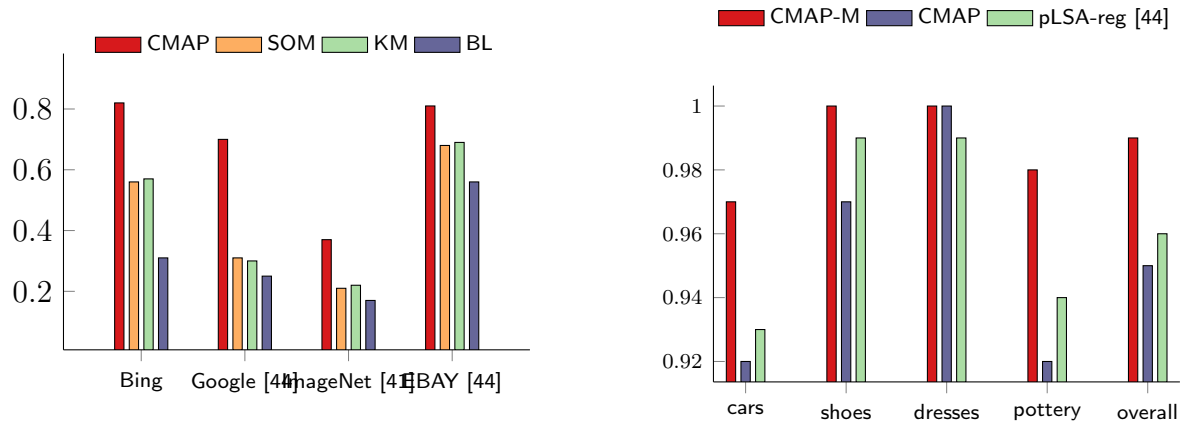


FIGURE 6: Left: Attribute recognition performances on novel images compared to other methods. **Right:** Equal Error Rates on EBAY dataset for image retrieval using the configuration of [44]. CMAP does not utilise the image masks used in [44], while CMAP-M does.

algorithm (**SOM**) with optimal cluster number, decided by cross-validation of whole pipeline, and again train a model for each cluster. The low results support the need for pruning of the data through outlier elimination. Results show that, CMAP's clusters are able to detect coherent and clean representative data groups so we train less number of classifiers by eliminating outlier clusters but those classifiers better in quality and also, on novel test sets with images having different characteristics than the images used in training, CMAP can still perform very well on learning of attributes.

Our method is also utilised for retrieving images on EBAY dataset as in [44]. [44] learns the models from web images and apply the models to another set so both method study a similar problem. We utilise CMAP with patches obtained from the entire images (**CMAP**) as well as from the masks provided by [44] (**CMAP-M**). As shown in Figure6 Right, even without masks CMAP is comparable to the performance of the PLSA based method of [44], and with the same setting CMAP outperforms the PLSA based method with significant performance difference.

On ImageNet, we obtained 37.4% accuracy compared to 36.8% of [41]. It is also seen that, our models trained from different source of information are better to generalize for some of worse performance classes (rough, spotted, striped, wood) of [41]. Recall that we globally learn the attribute models from web images, not from any partition of the ImageNet. Thus, it is encouraging to observe better results in such a large data-set against [41]'s attribute models trained by a sufficiently large training subset.

Attribute based scene recognition: While the results on different datasets support the ability of our approach to be generalised to different datasets, we also perform experiments to understand the effect of the learned attributes on a different task, namely for classification of scenes using entirely different collections. Experiments are performed on MIT-indoor [38], and Scene-15 [22] datasets. MIT-indoor has 67 different indoor scene with 15620 images with at least 100 images for each

category and we use 100 images from each class to test our results. Scene-15 is composed by 15 different scene categories. We use 200 images from each category for our testing. MIT-indoor is extended and even harder version of Scene-15 with many additional categories.

We again get the confidence values for each grid in three levels of the spatial pyramid on the test images. However, rather than using a single value for the maximum classifier output, we keep the confidence values for all the classifiers for each grid. We concatenate these vectors for all grids in all levels to get a single feature vector of size $3xNxK$ for the image, which is then used for scene classification. Here N is the number of grids at each level, and K is the number of different concepts. Note that, while the attributes are learned in an unsupervised way, in this experiment scene classifiers are trained on the datasets provided (see next section for automatic scene concept learning).

As shown in Table2, our method for scene recognition with learned attributes (**CMAP-A**), performs competitively with [24] while using shorter feature vectors in relatively cheaper environment, and outperforms the others. Comparisons with [38] show that using the visual information acquired from attributes is more descriptive in the cluttered nature of MIT-indoor scenes. For instance, "bookstore" images has very similar structural layout to "clothing store" images, but they are more distinct with colour and texture information around the scene. Attribute level features do not create this much difference for Scene-15 data-set since images include some obvious statistical differences.

3.1.5 LEARNING CONCEPTS FOR SCENE CATEGORIES

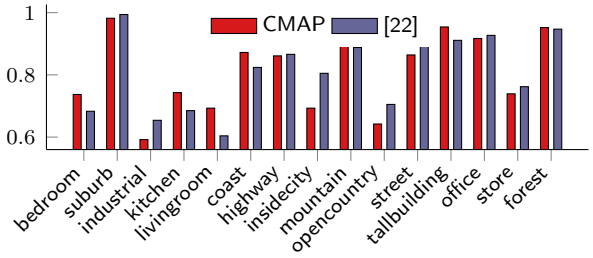
Alternative to recognising scenes through the learned attributes, we directly learn higher level concepts for scene categories. We call this method as **CMAP-S**. Specifically, we perform testing for scene classification for 15 scene categories on [22] and MIT-indoor [38] data-sets, but learn the scene concepts directly from the images collected from Web through querying for the names of the scene concepts used in these datasets. That is, we do not use any manually labelled training set (or training subset of the benchmark data-sets), but directly the crude web images which are pruned and organised by CMAP, in contrast to comparable fully supervised methods. As shown in Table2, our method is competitive with the state-of-the-art studies without requiring any supervised training.

We then made a slight change on our original CMAP-S implementation by using the hard-negatives of previous iteration as a negative set of next iteration (we refer to this new method as **CMAP-S-HM**). We relax the memory needs with less but strong negative instances. As the results in Table2 and Figure7 show, we achieve better performances in Scene-15 than the state-of-the-art studies with this simple addition, still without requiring any supervisory input. However, on a harder MIT-indoor dataset, without using attribute information, low-level features are not very distinctive.

In order to understand the effect of discriminative visual features, which aim to capture representative and discriminative mid-level features, we also compare our method with the work of Singh et al. [43]. As seen in Table2, our performances are better than both their reported results on MIT-indoor, and our implementation on Scene-15.

-	CMAP-A	CMAP-S	CMAP-S+HM	Li et al. [24] VQ	Pandey et al. [36]	Kwitt et al. [20]	Lazebnik et al. [22]	Singh et al. [43]
MIT-indoor [38]	46.2%	40.8%	41.7%	47.6%	43.1%	44%	-	38%
Scene-15 [22]	82.7%	80.7%	81.3%	82.1%	-	82.3%	81%	77%

TABLE 2: Comparison of our methods on scene recognition in relation to state-of-the-art studies on MIT-Indoor [38] and Scene-15 [22] datasets.



	CMAP	[11]	[23]		CMAP	[11]	[23]
airplane	0.63	0.51	0.76	car	0.97	0.98	0.94
face	0.67	0.52	0.82	guitar	**0.89**	0.81	0.60
leopard	0.76	0.74	0.89	motorbike	**0.98**	0.98	0.67
watch	**0.55**	0.48	0.53	overall	**0.78**	0.72	0.75

FIGURE 7: Left: Comparisons on Scene-15 dataset. Overall accuracy is 81.3% for CMAP-S+HM , versus 81% for [22] . Classes "industrial", "insidecity", "opencountry" results very noisy set of web images, hence trained models are not strong enough as might be observed from the chart. **Right:** Classification accuracies of our method in relation to [11] and [23].

3.1.6 LEARNING CONCEPTS OF OBJECT CATEGORIES

We learn object concepts from Google web images [11] and compare our results with [11] and [23] (Figure7 Right). [11] provides a data-set from Google with 7 classes and total 4088 gray scale images, 584 images in average for each class with many "junk" images in each class as they indicated. They test their results in a manually selected subset of Caltech Object data-set. Because of its raw nature of the Google images and adaptation to the Caltech subset, it is a good experimental ground for our pipeline.

Salient regions extracted from images are represented with 500 word quantized SIFT [26] vector with additional 256 dimension LBP [34] vector. In total we aggregated a 756 dimension vector representation for each salient region. At the final stage of learning with CMAP, we learn L2 norm, linear SVM classifiers for each cluster with negatives are gathered from other classes and the global outliers. For each learning iteration, we also apply hard mining to cull highest rank negative instances in the amount 10 times of salient instances in the cluster. All pipeline hyper-parameters are tuned via the validation set provided by [11]. Given a novel image, learned classifiers are passed over the image with gradually increasing scales, up to a point where the maximum class confidences are stable. Among class confidences, maximum confidence indicates the final prediction for that image. We observe 6.3 salient clusters in average for all classes and 69.4 instances for each salient clusters. That is, CMAP eliminates 147 instances for each class as supposedly outlier instances. Results support that elimination of "junk" images gives significant improvements, especially for the noisy classes in [11].

Method	GBC+CF(half)[35]	CMAP-1	CMAP-2	BaseLine
Easy	0.58	0.63	0.66	0.31
Hard	0.32	0.34	0.38	0.18

TABLE 3: Face learning results with detecting faces using OpenCV(CMAP-1) and [51](CMAP-2).

3.1.7 LEARNING FACES

We use FAN-large [35] face data-set for testing our method in face recognition problem. We use Easy and Hard subsets with the names accommodating more than 100 images (to have fair testing results). Our models are trained over web images queried from Bing Image search engine for the same names. All the data preprocessing and the feature extraction flow follow the same line of [35], that is owned from [10]. However, [35] trains the models and evaluates the results at the same collection.

We retrieve the top 1000 images from Bing results. Face are detected and face with the highest confidence is extracted from each image to be fed into CMAP. Face instances are clustered and spurious face instances are pruned. Salient clusters are used for learning SVM models for each cluster in the same settings of the object categories. For our experiments we used two different face detectors. One is cascade classifier of [46] implemented in OpenCV library [2] and another is [51] with more precise detection results, even the OpenCV implementation is very fast relatively. Results are depicted at Table3 with two different face detection method and baseline result with models trained on raw Bing images for each person.

4 CONCEPTUAL INDEXING

Acknowledging the diversity of concepts and concept representations, the MUCKE Framework implements a flexible concept package, shown in Figure 8. Every indexed document consists of one or more types of facets: TextFacets, TagFacets and ImageFacets corresponding to the different types of data a document can contain. By processing these fields, a list of concept objects is obtained.

A concept can be of either two types: textual and visual. These two types, however, are united in one concept class and distinguished with a type attribute. Every concept has an unique identifier and an optional link to a File that contains a classifier (e.g. a Markov model or a neural network). Such models are too complex to express and standardize for all possible cases and are therefore externalized. It makes also very little sense to include their structure in a Java class since their processing will most likely not be performed in a uniform development environment but more likely with specific languages and environments (e.g. R, SPSS, Matlab, Mathematica). If there is no implicit classifier model that describes the concept, then this file link is simply set to null. This will be the most common case.

As defined in the S3 meeting in Paris, a concept will most likely point to a Wikipedia article. Should a concept not exist, then it can be created for the refined scope of the project. Is the

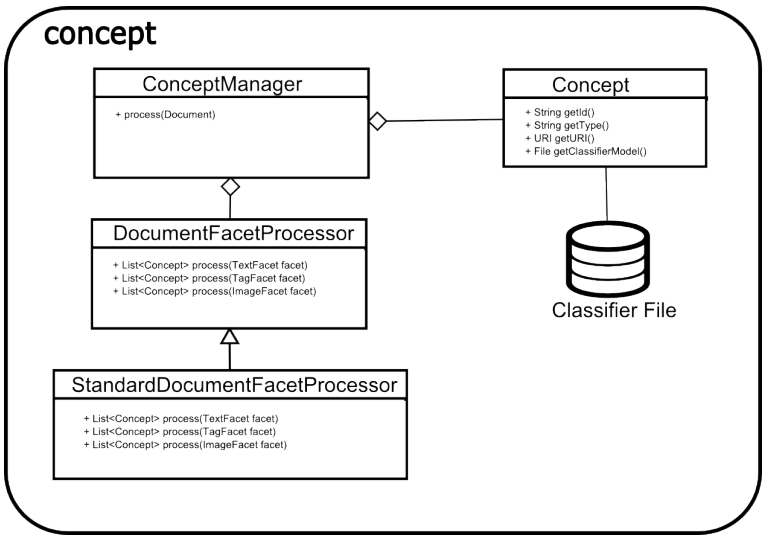


FIGURE 8: Concept architecture in the MUCKE Framework

concept very fine-grained and hard to pin down (e.g. a particular texture of oak wood), then is shall be defined by pointing to the concept "oak" or "oak texture" and further refined by linking to a classifier model that identifies this particular type of oak texture. With this combination, it is possible to define very fine differences.

In order to extract concepts from facets, the visitor pattern has been used which allows the three document facets (for text, tag and image facets) to have alternative implementations of how to identify concepts.

5 CONCLUSIONS

In the context of the MUCKE project multimedia fusion is primarily done via mapping both text and images to a conceptual space. The nature of this conceptual space is complex and the MUCKE Framework has been designed to be able to map different instances and types of concepts. The direct integration between images and text has already shown its benefits, as described in Section **??**[2]. Images and text have also individually benefitted from being raised to a higher semantic level, as described in Sections 3 and 2, respectively. The MUCKE project continues its work on concept integration, towards the final prototype at the end of the upcoming year.

6 ADDITIONAL AUTHORS

- Navid Rekabsaz, Vienna University of Technology
- Phong Vo, Alexandru Ginsca, Hervè Le Borgne, CEA LIST

[2]not available in the public version, as it is pending review

References

[1] R. Bierig, C. Serban, A. Siriteanu, M. Lupu, and A. Hanbury. A System Framework for Concept- and Credibility-Based Multimedia Retrieval. In *Proc. of ICMR*, 2014.

[2] G. Bradski. Opencv library. *Dr. Dobb's Journal of Software Tools*, 2000.

[3] Burges, C. J. C., Ragno, Robert, Le, and Q. V. Learning to rank with nonsmooth cost functions. MIT Press, 2006.

[4] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. Learning to rank using gradient descent. In *Proc. of ICML*, 2005.

[5] Z. Cao, T. Qin, T.-Y. Liu, M.-F. Tsai, and H. Li. Learning to rank: From pairwise approach to listwise approach. In *Procs of ICML*, 2007.

[6] S. Clinchant, J. Ah-Pine, and G. Csurka. Semantic combination of textual and visual information in multimedia retrieval. In *Proceedings of the 1st ACM International Conference on Multimedia Retrieval*, pages 44:1–44:8. ACM, 2011.

[7] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In *Workshop on Statistical Learning in Computer Vision (at ECCV)*, 2004.

[8] A. Depeursinge and H. Müller. Fusion techniques for combining textual and visual information retrieval. In H. Müller, P. Clough, T. Deselaers, and B. Caputo, editors, *ImageCLEF*, volume 32, pages 95–114. Springer Berlin Heidelberg, 2010.

[9] E. Erdem and A. Erdem. Visual saliency estimation by nonlinearly integrating features using region covariances. *Journal of Vision*, 13(4):1–20, 2013.

[10] M. Everingham, J. Sivic, and A. Zisserman. Hello! my name is... buffy–automatic naming of characters in tv video. 2006.

[11] R. Fergus, L. Fei-Fei, P. Perona, and A. Zisserman. Learning object categories from google's image search. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 2, pages 1816–1823. IEEE, 2005.

[12] Y. Freund, R. Iyer, R. E. Schapire, and Y. Singer. An efficient boosting algorithm for combining preferences, 1998.

[13] L. I. Hang. A short introduction to learning to rank. *Transactions on Information and Systems*, 2011.

[14] D. H. Hubel and T. N. Wiesel. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology*, 160(1):106, 1962.

[15] S. J. Hwang and K. Grauman. Accounting for the relative importance of objects in image retrieval. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2010.

[16] S. Jonnalagadda, T. Cohen, S. Wu, and G. Gonzalez. Enhancing clinical concept extraction with distributional semantics. *Journal of Biomedical Informatics*, 45(1):129 – 140, 2012.

[17] J. Karlgren and M. Sahlgren. From words to understanding. In Y. Uesaka, P. Kanerva, and H. Ashton, editors, *Foundations of Real-World Intelligence*. 2001.

[18] T. Kohonen. *Self-organizing maps*. Springer, 1997.

[19] T. Kolenda, L. Hansen, J. Larsen, and O. Winther. Independent component analysis for understanding multimedia content. In *Neural Networks for Signal Processing, 2002. Proceedings of the 2002 12th IEEE Workshop on*, pages 757–766, 2002.

[20] R. Kwitt, N. Vasconcelos, and N. Rasiwasia. Scene recognition on the semantic manifold. *ECCV*, 2012.

[21] T. K. Landauer and S. T. Dumais. A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological review*, pages 211–240, 1997.

[22] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 2169–2178. IEEE, 2006.

[23] L.-J. Li and L. Fei-Fei. Optimol: automatic online picture collection via incremental model learning. *International journal of computer vision*, 88(2):147–168, 2010.

[24] Q. Li, J. Wu, and Z. Tu. Harvesting mid-level visual concepts from large-scale internet images. *CVPR*, 2013.

[25] T.-Y. Lie. *Learning to rank for information retrieval. Foundations and Trends in Information Retrieval*. Springer, 2011.

[26] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[27] K. Lund and C. Burgess. Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior Research Methods*, 28, 1996.

[28] M. Lupu. On the Usability of Random Indexing in Patent Retrieval. In *Proc. of ICCS*, 2014.

[29] C. Macdonald, R. L. Santos, and I. Ounis. The whens and hows of learning to rank for web search. *Inf. Retr.*, 16(5), 2013.

[30] J. a. Magalhães and S. Rüger. Information-theoretic semantic multimedia indexing. In *Proceedings of the 6th ACM international conference on Image and video retrieval*, pages 619–626. ACM, 2007.

[31] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[32] A. Muñoz and J. Muruzábal. Self-organizing maps for outlier detection. *Neurocomputing*, 18(1):33–60, 1998.

[33] H. Nottelmann and N. Fuhr. From retrieval status values to probabilities of relevance for advanced ir applications. *Information Retrieval*, 6, 2003.

[34] T. Ojala, M. Pietikainen, and T. Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(7):971–987, 2002.

[35] M. Özcan, J. Luo, V. Ferrari, and B. Caputo. A large-scale database of images and captions for automatic face naming. In *BMVC*, pages 1–11, 2011.

[36] M. Pandey and S. Lazebnik. Scene recognition and weakly supervised object localization with deformable part-based models. *ICCV*, 2011.

[37] M. L. Paramita and M. Grubinger. Photographic image retirval. In H. Müller, P. Clough, T. Deselaers, and B. Caputo, editors, *ImageCLEF: Experimental Evaluation in Visual Information Retrieval*, pages 141–162. Springer, 2010.

[38] A. Quattoni and A. Torralba. Recognizing indoor scenes. *CVPR*, 2009.

[39] N. Rasiwasia, J. Costa Pereira, E. Coviello, G. Doyle, G. R. Lanckriet, R. Levy, and N. Vasconcelos. A new approach to cross-modal multimedia retrieval. In *Proceedings of the international conference on Multimedia*, pages 251–260. ACM, 2010.

[40] N. Rekabsaz and M. Lupu. A real-world framework for translator as expert retrieval. In *Proc. of CLEF*, 2014.

[41] O. Russakovsky and L. Fei-Fei. Attribute learning in large-scale datasets. In *Trends and Topics in Computer Vision*, pages 1–14. Springer, 2012.

[42] H. Schütze. Dimensions of meaning. In *Supercomputing '92. Proceedings*, 1992.

[43] S. Singh, A. Gupta, and A. A. Efros. Unsupervised discovery of mid-level discriminative patches. In *Computer Vision–ECCV 2012*, pages 73–86. Springer, 2012.

[44] J. Van De Weijer, C. Schmid, J. Verbeek, and D. Larlus. Learning color names for real-world applications. *Image Processing, IEEE*, 2009.

[45] K. van Rijsbergen. *The Geometry of Information Retrieval*. Cambridge University Press, 2004.

[46] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–511. IEEE, 2001.

[47] J. Wang, D. Song, and L. Kaliciak. Tensor product of correlated text and visual features: A quantum theory inspired image retrieval framework. In *AAAI-Fall 2010 Symposium on Quantum Informatics for Cognitive, Social, and Semantic Processes (QI'2010)*, pages 109–116, 2010.

[48] Q. Wu, C. J. C. Burges, K. M. Svore, and J. Gao. Adapting boosting for information retrieval measures. Springer Science, 2009.

[49] J. Xu and H. Li. Adarank: A boosting algorithm for information retrieval. In *Procs of SIGIR*, 2007.

[50] B. Q. Zadeth and S. Handschuh. Evaluation of Technology Term Recognition with Random Indexing. In *Proc. of LREC* [33].

[51] X. Zhu and D. Ramanan. Face detection, pose estimation, and landmark localization in the wild. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2879–2886. IEEE, 2012.