Two-step relevance feedback for semantic disambiguation in image retrieval

Daniel Heesch Stefan Rueger

PIXSTA

PICTURE YOUR SEARCH

Outline

- Content based image search
- Relevance Feedback
- The NN^k Idea
- Experiments and Results
- Conclusions

Content based image search

- Using an external image
  e.g. bigimbaz, retrievr
- Using an internal image
  e.g. pixsta
- Using keywords (and automated image annotation)
  e.g. behold

A key question: how to measure similarity

- How to select/weight visual features

A key question: how to measure similarity

- How to select/weight visual features
- Maximise retrieval performance for
  - ...a particular collection.
  - ...this particular query image.
  - ...this particular user.
  - ...this particular semantic facet of this particular query.

A key question: how to measure similarity

- How to select/weight visual features
- Maximise retrieval performance for
  - ...a particular collection.
  - ...this particular query image.
  - ...this particular user.
  - ...this particular semantic facet of this particular query.
- Different features capture different semantic facets




A key question: how to measure similarity

- How to select/weight visual features
- Maximise retrieval performance for
  - ...a particular collection.
  - ...this particular query image.
  - ...this particular user.
  - ...this particular semantic facet of this particular query.
- Different features capture different semantic facets

- Of course, some features don't capture anything meaningful, and some facets are not captured by any features

Relevance feedback

Endow the retrieval system with some degree of freedom, e.g. a parametrised metric

1. Retrieve a first set of results with the system's default state.
2. Collect relevance feedback by letting users mark relevant and non-relevant images.
3. Change the system state, e.g. update parameters of the metric.
4. Retrieve again and go to 2 until user is satisfied.

Relevance feedback

- Many realisations of this basic template, e.g.
  - Rui *et al.* 98: $w_i \propto \frac{1}{\sigma_i}$
  - Ishikawa *et al.* 98: compute optimal query vector from positive and negative examples
  - Tong *et al.* 01: estimation of hyperplane between relevant and non-relevant images
  - Urban *et al.* 03: query weighted average over relevant images
  - Giacinto *et al.* 04: non-parametric instance-based relevance feedback
  - plus 400+ more

Relevance feedback

- Many realisations of this basic template, e.g.
  - Rui *et al.* 98: $w_i \propto \frac{1}{\sigma_i}$
  - Ishikawa *et al.* 98: compute optimal query vector from positive and negative examples
  - Tong *et al.* 01: estimation of hyperplane between relevant and non-relevant images
  - Urban *et al.* 03: query weighted average over relevant images
  - Giacinto *et al.* 04: non-parametric instance-based relevance feedback
  - plus 400+ more
- Limitations: default setting may not be suitable for the semantic facet of interest. In the worst case, no relevant images are retrieved in the first step

The NN^k idea

- The agnostic stance: assume we know nothing about which features may be important or which semantic facet a user is interested in.
- We extract the query's semantic facets by determining its neighbours under many different feature combinations.
- For each feature combination, we record only the nearest neighbour.
- There may not be many relevant nearest neighbours, but, if the features are any good, the semantic facet of interest should be represented by at least one neighbour.

The NN^k idea

- Determine for a query image q all those images p from collection $\mathcal{I}$ that are closest under *some* metric out of a family of metrics f_w.

The NN^k idea

- Determine for a query image q all those images p from collection $\mathcal{I}$ that are closest under *some* metric out of a family of metrics f_w.
- Formally, p is an NN^k if, for some w,

$$p = \arg\min_{i \in \mathcal{I}} f_w(i, q),$$

The NN^k idea

- Determine for a query image q all those images p from collection $\mathcal{I}$ that are closest under *some* metric out of a family of metrics f_w.
- Formally, p is an NN^k if, for some w,

$$p = \arg\min_{i \in \mathcal{I}} f_w(i, q),$$

- We assume

$$f_w(x, y) = \sum_{i=1}^{k} w_i d_i(x, y) \quad \text{(Why?)}$$

where $d(\cdot, \cdot)$ are feature specific distance functions.

The NN^k idea

- Determine for a query image q all those images p from collection $\mathcal{I}$ that are closest under *some* metric out of a family of metrics f_w.
- Formally, p is an NN^k if, for some w,

$$p = \arg\min_{i \in \mathcal{I}} f_w(i, q),$$

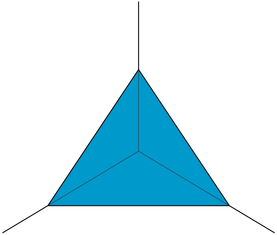
- We assume

$$f_w(x, y) = \sum_{i=1}^{k} w_i d_i(x, y) \quad \text{(Why?)}$$

  where $d(\cdot, \cdot)$ are feature specific distance functions.
- We further assume that the w's add to 1 and are positive (Why?)

The NN^k idea with $k = 3$

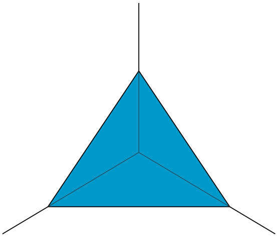
- With $k = 3$, all weight vectors lie on a 2-dimensional simplex



- For different positions on the simplex, we may get different nearest neighbour of an image query

The NN^k idea with $k = 3$

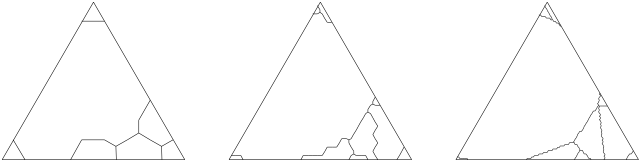
- With $k = 3$, all weight vectors lie on a 2-dimensional simplex



- For different positions on the simplex, we may get different nearest neighbour of an image query
- What is the simplex for $k = 2$?

The NN^k idea with $k = 3$

- Finding the NN^k by discretising the weight space.



- Each NN^k can be associated with a typical weight vector under which it is retrieved. We choose the mean weight vector.

The NN^k idea for relevance feedback

The proposed relevance feedback method

1. Retrieve the set of NN^k and determine supporting feature weights
2. Let users select relevant images from the set
3. Retrieve again with the selected weights

Experiments - Image data and visual features

- Two collections
  - Corel images: 32,000 images, 191 classes, 1.1 avg polysemy
  - Getty images: 8,000 images, 100 classes, 1.4 avg polysemy

Experiments - Image data and visual features

- Two collections
  - Corel images: 32,000 images, 191 classes, 1.1 avg polysemy
  - Getty images: 8,000 images, 100 classes, 1.4 avg polysemy
- Eight features ($\rightarrow NN^8$)
  - Tamura texture features
  - Gabor wavelets
  - HSV histograms
  - Colour Structure Descriptor
  - ...and others

Experiments - Benchmarking

- Uniform weights, i.e. $1/k$

Experiments - Benchmarking

- Uniform weights, i.e. $1/k$
- Oracle: performance for best weight vector found by gradient ascent on k-dimensional performance landscape

Experiments - Benchmarking

- Uniform weights, i.e. $1/k$
- Oracle: performance for best weight vector found by gradient ascent on k-dimensional performance landscape
- Feature selection method by Rui et al. (2000)

1. Retrieve with uniform weights
2. Select and score relevant images
3. Update the weight of the jth feature according to

$$w_j \propto \left(\sum_{i=1}^{N} v_i d(p_{ij}, q_j) \right)^{-\frac{1}{2}}$$

where the sum is over the set of relevant images. This minimises the sum of the v_i-weighted distances between relevant images and the query.

Experimental details

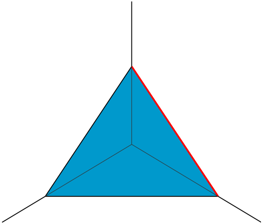
- One ranked list for each relevant NN^k. How to merge these?
  - Round robin: ranks items according to their highest rank
  - CombSum: rank items according to average score
  - Borda fuse: rank items according to average rank

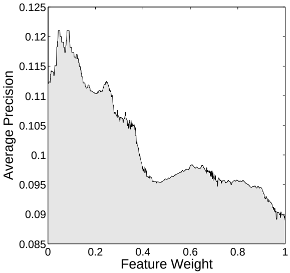
Experimental details

- One ranked list for each relevant NN^k. How to merge these?
  - Round robin: ranks items according to their highest rank
  - CombSum: rank items according to average score
  - Borda fuse: rank items according to average rank
- Performance measures:
  - Precision at 50 (P50)
  - Mean Average Precision (MAP)

Experimental details

- One ranked list for each relevant NN^k. How to merge these?
  - Round robin: ranks items according to their highest rank
  - CombSum: rank items according to average score
  - Borda fuse: rank items according to average rank
- Performance measures:
  - Precision at 50 (P50)
  - Mean Average Precision (MAP)
- Determining oracle performance





Results

	Corel		Getty	
	MAP (%)	Pr50(%)	MAP (%)	Pr50 (%)

Results

	Corel		Getty	
	MAP (%)	Pr50(%)	MAP (%)	Pr50 (%)
Baseline	2.29	1.58	2.95	1.64

Results

	Corel		Getty	
	MAP (%)	Pr50(%)	MAP (%)	Pr50 (%)
Baseline	2.29	1.58	2.95	1.64
Oracle	5.26	2.81	6.41	2.92

Results

	Corel		Getty	
	MAP (%)	Pr50(%)	MAP (%)	Pr50 (%)
Baseline	2.29	1.58	2.95	1.64
Oracle	5.26	2.81	6.41	2.92
RUI	2.49	1.66	3.17	1.71

Results

	Corel		Getty	
	MAP (%)	Pr50(%)	MAP (%)	Pr50 (%)
Baseline	2.29	1.58	2.95	1.64
Oracle	5.26	2.81	6.41	2.92
RUI	2.49	1.66	3.17	1.71
NN^k (BF)	3.43	1.83	4.55	2.02

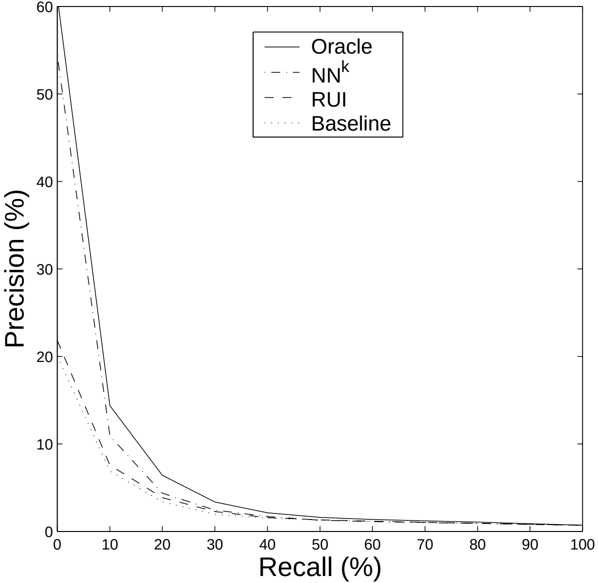
Results

	Corel		Getty	
	MAP (%)	Pr50(%)	MAP (%)	Pr50 (%)
Baseline	2.29	1.58	2.95	1.64
Oracle	5.26	2.81	6.41	2.92
RUI	2.49	1.66	3.17	1.71
NN^k (BF)	3.43	1.83	4.55	2.02
NN^k (CS)	3.79	1.95	4.91	2.14

Results

	Corel		Getty	
	MAP (%)	Pr50(%)	MAP (%)	Pr50 (%)
Baseline	2.29	1.58	2.95	1.64
Oracle	5.26	2.81	6.41	2.92
RUI	2.49	1.66	3.17	1.71
NN^k (BF)	3.43	1.83	4.55	2.02
NN^k (CS)	3.79	1.95	4.91	2.14
NN^k (RR)	4.45	2.12	5.76	2.28

Results



Oracle
NN^k
RUI
Baseline
Precision (%)
Recall (%)
0
10
20
30
40
50
60
70
80
90
100

Conclusions & Questions

- Simple framework for relevance feedback that shows good performance on fairly realistic datasets
- Individual distance functions and features can be more complex, e.g. EMD on keypoint representations (Jeong & Grauman, 2008)
- How can this two-step (one-shot) method be extended to several rounds of feedback?