CSE 6363 Presentation

Lin Sun, 1001855171

2021/11/22

SIMON TONG, EDWARD CHANG

ACM MULTIMEDIA CONFERENCE, 107-118, 2001

SUPPORT VECTOR MACHINE ACTIVE LEARNING FOR IMAGE RETRIEVAL

CONTENTS

IMAGE RETRIEVAL

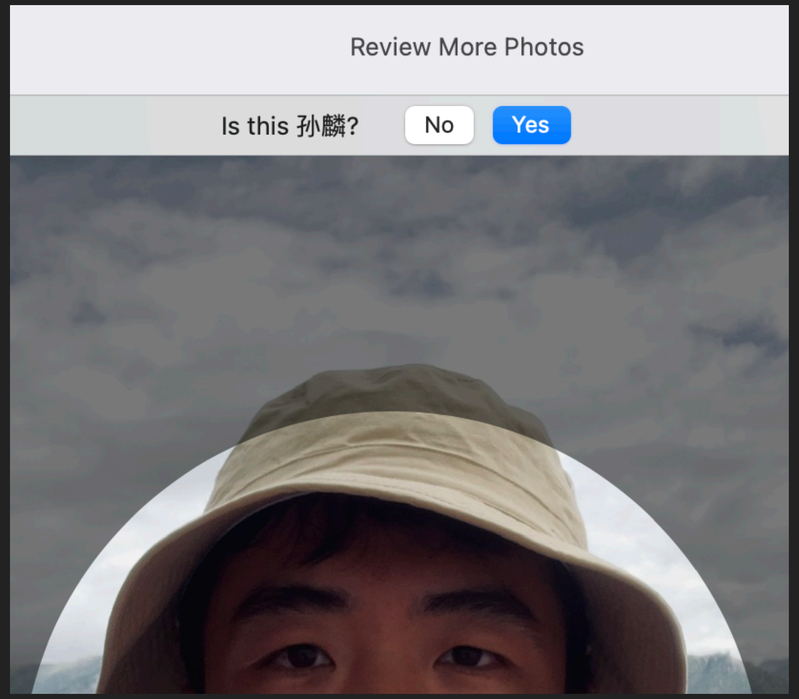
- Based on a query, feedback relevant images in an image database.
  - images.google.com
- Query:
  - Keywords, Images
- Results:
  - List of images, ordered by relevance/similarity
- Performance metrics
  - Precision @Top N

ACTIVE LEARNING

- Active Learning: aims to select the most informative examples for labeling from the pool.
- Pool-based:
  - A pool of unlabeled data: the entire database of images —– in this case
  - Request the user’s label for a certain number of instances in the pool
    - Two possible labeling of an image: relevant or non-relevant
- Goal: to learn the user’s query concept —– query refinement —– in this case

ACTIVE LEARNING EXAMPLE

- Image Classification: people classification in image applications
  - Google Photos
  - iCloud Photos



CONTENTS

PURPOSE

- Using a **S**upport **V**ector **M**achine **A**ctive **L**earning algorithm for conducting effective **relevance feedback** of **I**mage **R**etrieval
  - selects the most informative images to query a user
  - and quickly learns a boundary that separates the images
    - that satisfy the user’s query concept
    - from the rest of the dataset
- **Query Refinement**

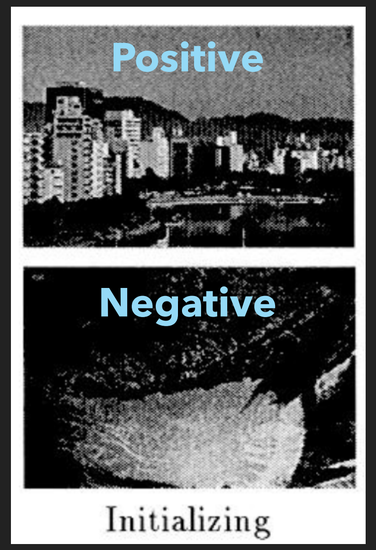
QUERY REFINEMENT

- Given an initial query, extends the query to a more specific concept
  - Non-interactive
    - Hierarchical dictionary, Labels, Semantics
  - Interactive
    - **Relevance feedback**
    - Analyzing clicks in the results, user behavior analysis
- Ultimate Goal: to improve the performance of image retrieval

RELEVANCE FEEDBACK

Learning

1

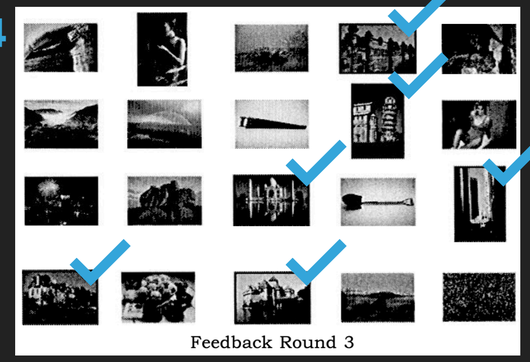


Query

2

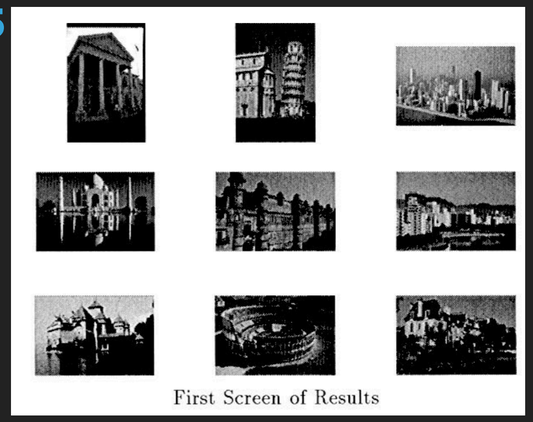
3

4

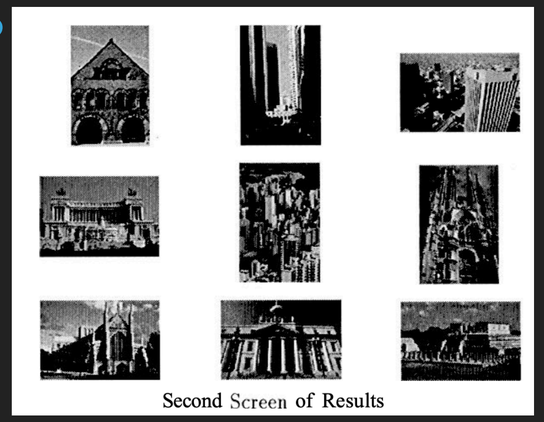


Retrieval

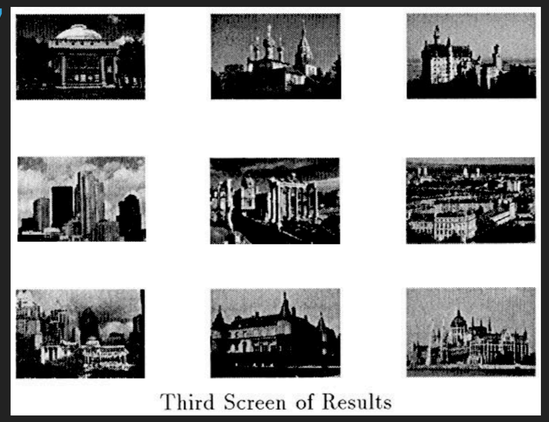
5



6



7



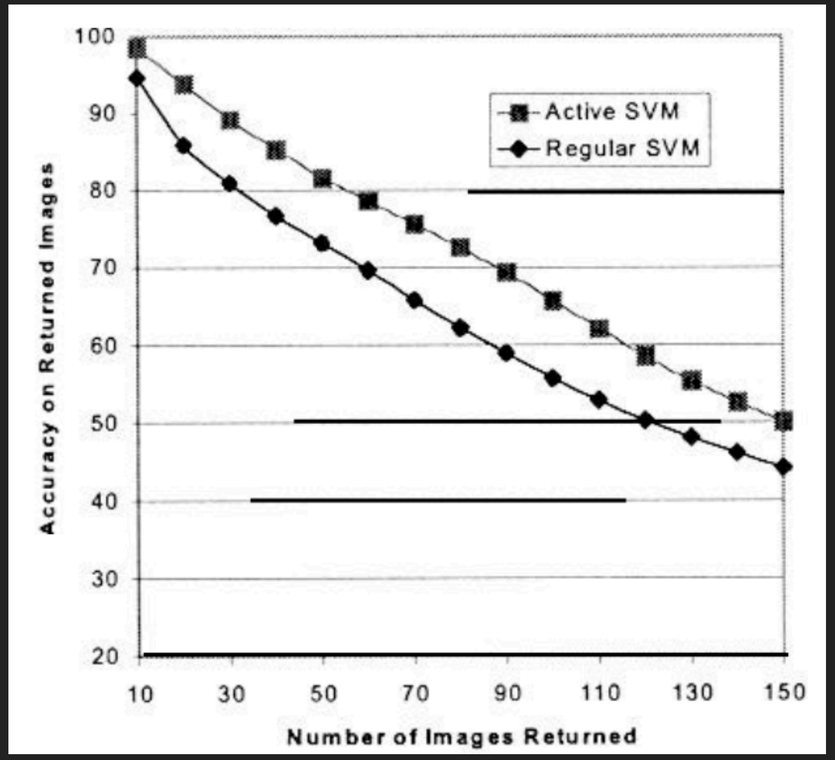
CONTENTS

ACTIVE VS PASSIVE

- Pool-query: choose informative images within the pool to ask the user to label
  - Most ML algorithms are **passive**: using a **randomly** selected training set
  - Active Learning: choosing its next pool-query **based upon the past answers** to previous pool-queries

BETTER PERFORMANCE

- Design goals:
  - Learn target concept **accurately:**
    - Given the same rounds of feedbacks, active learning should result in higher accuracy of retrieval
  - Grasp a concept **quickly**, with only a small number of labeled instances
    - To achieve the same level of accuracy, active learning should require fewer rounds of feedbacks



Desired effect (accuracy)

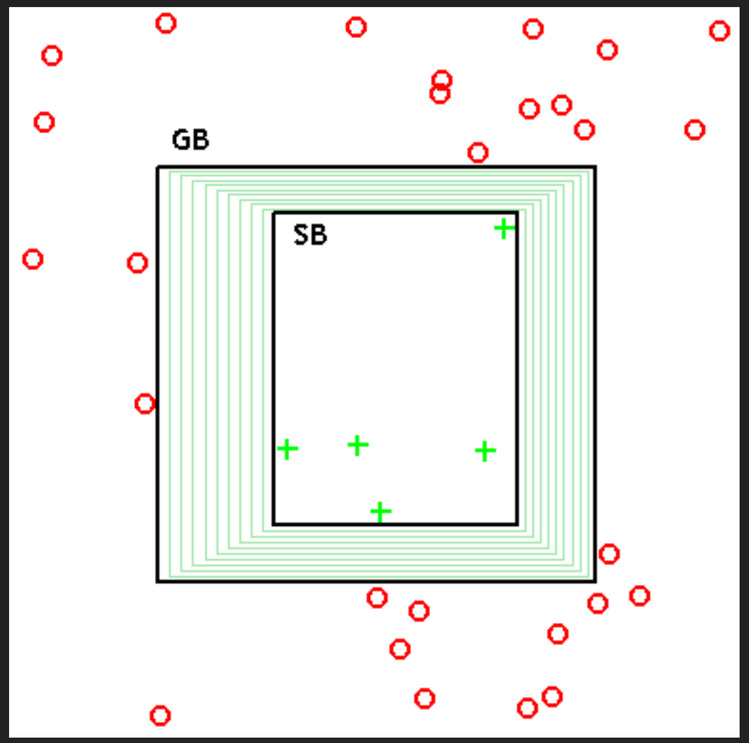
CONTENTS

HOW ACTIVE CAN BE BETTER THAN PASSIVE?

- Lemma (Tong & Koller, 2000)
  - $$\forall i \in \mathbb{N}^{+} \quad \sup_{P \in \mathcal{P}} E_P[Area(\mathcal{V}_i^*)] \leq \sup_{P \in \mathcal{P}} E_P[Area(\mathcal{V}_i)],$$
  - Active learning always queries instances whose corresponding hyperplanes in **parameter space** W halves the area of current version space
  - It always minimizes the maximum expected size of the version space
  - $Area(V)$ is the surface area that the version space V occupies on the hypersphere $\|\mathbf{w}\| = 1$

VERSION SPACE LEARNING

- Search a predefined space of hypotheses, viewed as a set of logical sentences (binary classification) [wikipedia]
  - GB is the maximally general positive hypothesis boundary
  - SB is the maximally specific positive hypothesis boundary
  - The intermediate rectangles represent the hypotheses in the version space

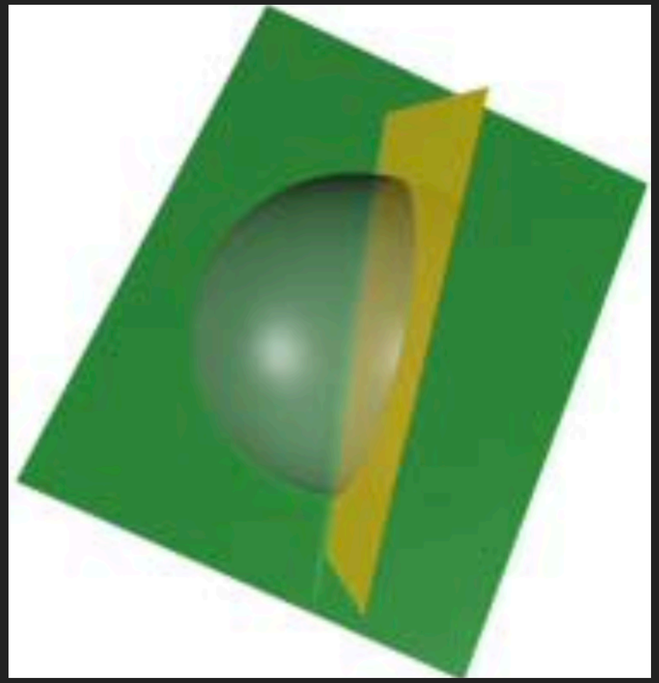


SVM SPACES

- **Input space**: $\mathcal{X} \subseteq \mathbb{R}^d$ contains training data $\{\mathbf{x}_1 \dots \mathbf{x}_n\}$
  - with labels $\{y_1 \dots y_n\}$ where $y_i \in \{-1, 1\}$
- **Feature space**: $\mathcal{F}$ is derived from input space via kernel $K(\mathbf{u}, \mathbf{v}) = \Phi(\mathbf{u}) \cdot \Phi(\mathbf{v})$ as $\Phi(\mathbf{x})$
- **Parameter space**: $\mathcal{W}$ is in classifier $f(\mathbf{x}) = \left(\sum_{i=1}^{n} \alpha_i K(\mathbf{x}_i, \mathbf{x}) \right) = \mathbf{w} \cdot \Phi(\mathbf{x}), \text{ where } \mathbf{w} = \sum_{i=1}^{n} \alpha_i \Phi(\mathbf{x}_i).$
  - SVM computes $\{\alpha_i\}$ the correspond to the maximal margin hyperplane in $\mathcal{F}$

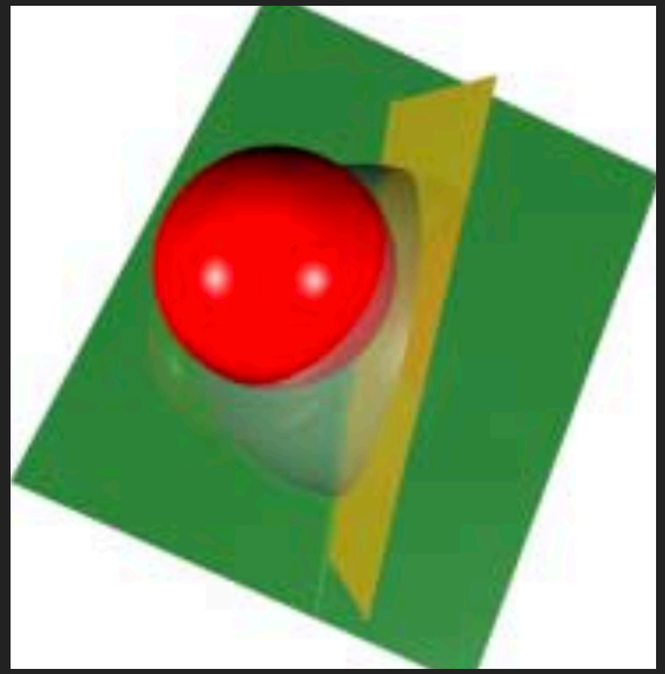
VERSION SPACE

- **Version Space** is a set of hyperplanes (hypotheses) that separate the data in the induced **Feature Space**
- The set of possible hypotheses: $$\mathcal{H} = \left\{ f \mid f(\mathbf{x}) = \frac{\mathbf{w} \cdot \Phi(\mathbf{x})}{\|\mathbf{w}\|} \ \textit{where}\ \mathbf{w} \in \mathcal{W} \right\},$$
- Version Space: $$\mathcal{V} = \{\mathbf{w} \in \mathcal{W} \mid \|\mathbf{w}\| = 1,\ y_i(\mathbf{w} \cdot \Phi(\mathbf{x}_i)) > 0, i = 1 \ldots n\}$$
  - Only exists if the training data are linearly separable in the feature space
- Parameter space $\mathcal{W}$ equal to feature space $\mathcal{F}$ because $$\mathbf{w} = \sum_{i=1}^{n} \alpha_i \Phi(\mathbf{x}_i)$$
- There is a bijection between unit vectors w and hypotheses f in H (hyperplane in Feature Space)
  - w is the normal vector of a hyperplane in F



WHY HALVES THE AREA OF VERSION SPACE?

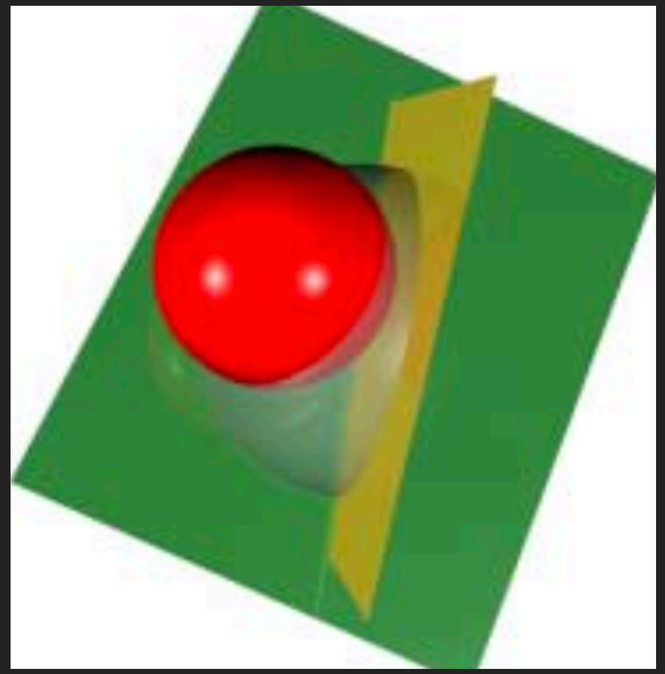
- SVMs is to find the point w in Version Space that maximize the margin, i.e., $\min_i\{y_i(\mathbf{w} \quad \Phi(\mathbf{x}_i))\}$, subject to $\|\mathbf{w}\| = 1$ and $y_i(\mathbf{w} \ \Phi(\mathbf{x}_i)) > 0 \quad i = 1.. \quad .n$
  - Consider points in F correspond to hyperplanes in W (F and W are identical)
  - $y_i\Phi(\mathbf{x}_i)$ as being the normal vector of a hyperplane in W (RBF kernel where $\|\Phi(\mathbf{x}_i)\| = \lambda$)
  - $y_i(\mathbf{w} \cdot \Phi(\mathbf{x}_i)) = \mathbf{w} \cdot y_i\Phi(\mathbf{x}_i) > 0$ defines a half-space in W
  - Thus we want to find the point in version space that maximizes the margin (minimum distance) to any of the delineating hyperplanes
  - That is the center of the largest radius hypersphere whose center can be placed in version space and whose surface does not intersect with the hyperplanes corresponding to the labeled instances
- Binary search is the most efficient way to reduce the size of version space without additional info



CONTENTS

HOW TO HALF THE AREA OF VERSION SPACE

- Select the unlabeled instances that split the current version space into two equal parts as much as possible
  - It is impractical to explicitly compute the sizes of the new version spaces V+ and V-
  - Ways to approximate
    - **Simple Margin: assume w is centrally placed in version space**
    - MaxMin Margin, Ratio Margin
      - calculate the margins (m+ and m-) for every unlabeled instance
      - too computation intensive (at the time of 2001), unsuitable for user-facing application

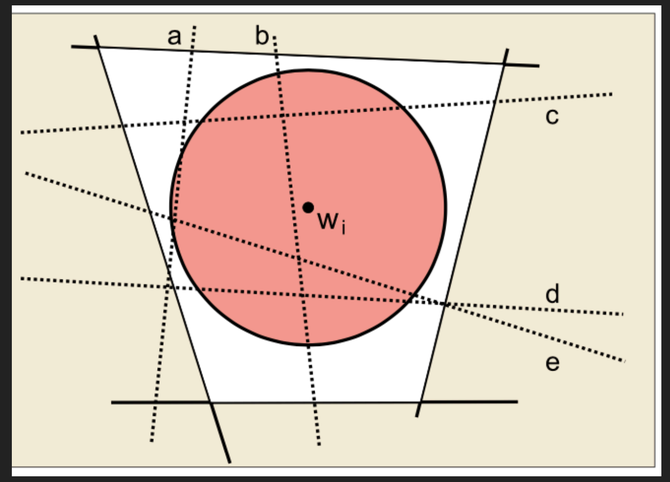


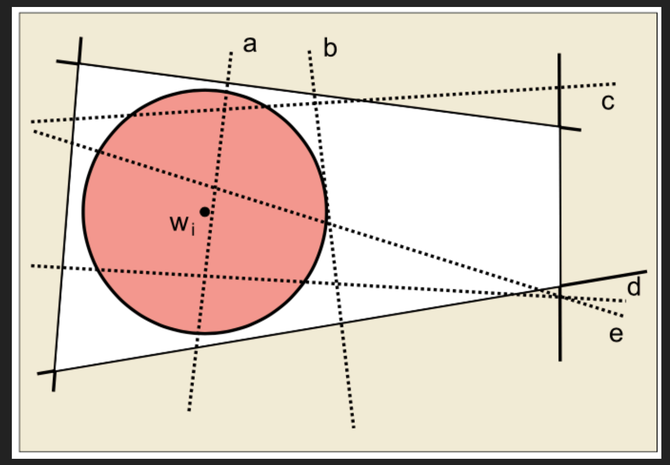
SINGLE MARGIN

- Select the unlabeled instances that split the current version space into two equal parts as much as possible
  - Best to select the hyperplanes exactly go through the center of the version space
  - Or, at least select the most closest ones
  - Then, how to identify the center?
- Single-Margin assumes each time the **learned** position of $\mathbf{w}_i$ is approximately in the center of the version space (although not always, but simple and fast)

SINGLE MARGIN

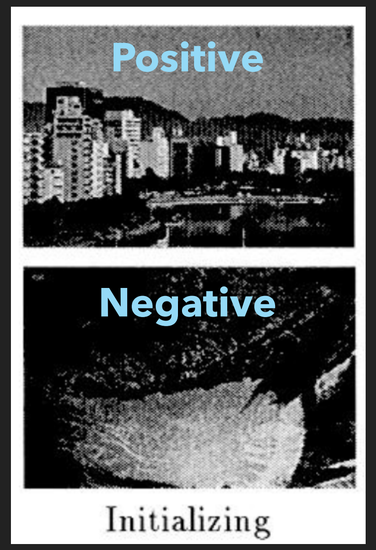
- Since $\mathbf{w}_i$ is considered as the center of current version space
  - The shortest distance between $\mathbf{w}_i$ and
    - the hyperplane corresponds to each instance x in the parameter space $\mathcal{W}$
  - Is the distance between the hyperplane $\mathbf{w}_i$ in the feature space $\mathcal{F}$ and
    - the feature vector $\Phi(\mathbf{x})$
  - which is as easy as $|\mathbf{w}_i \quad \Phi(\mathbf{x})|$
- In practice, during each round the closest **N=20** images will be selected
  - The first round uses random-selection due to the unstable issue of single-margin





ALGORITHM

1



Query

Round 1 (passive)

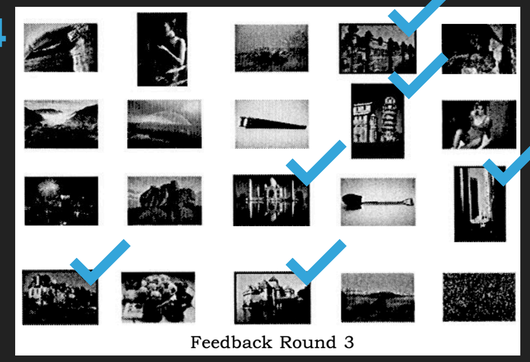
2

Round 2 (active)

3

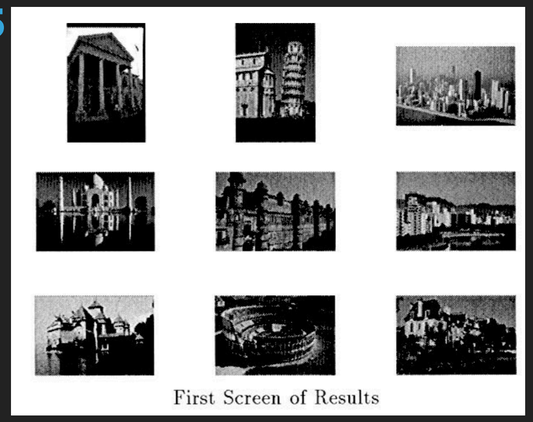
Round 3 (active)

4

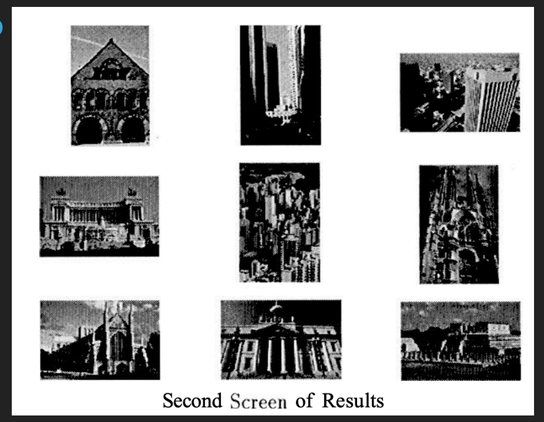


Retrieval (top-k)

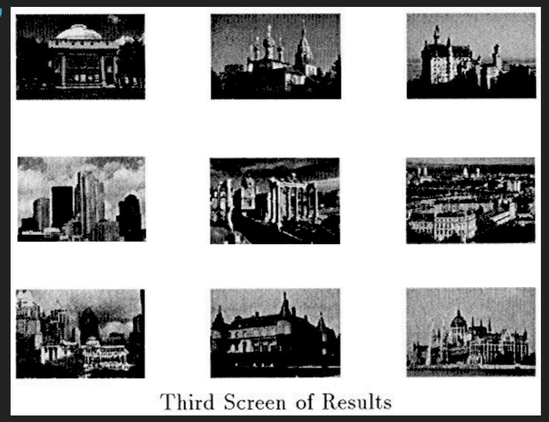
5



6



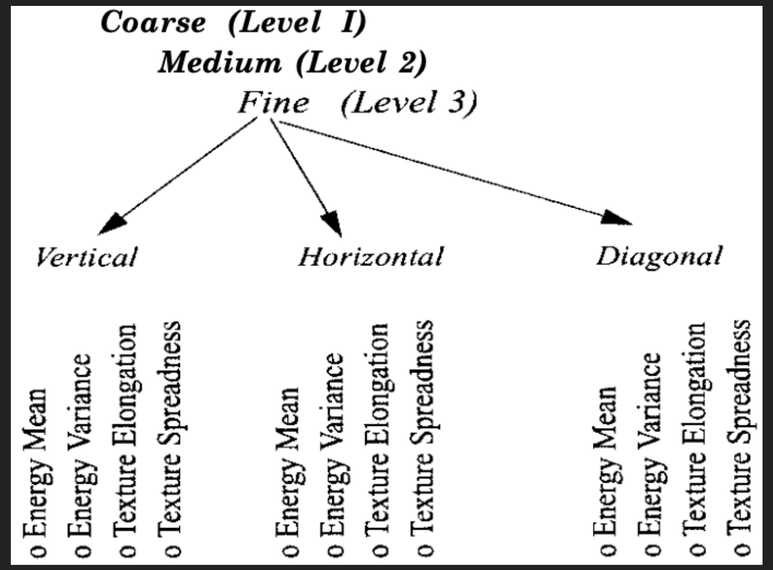
7



INPUT SPACE

- 144-dimension:
  - Color
    - 12-bit color mask, spreadness (scatter), elongation (shape)
    - color histogram, HSV means, HSV variances
  - Texture
    - 9 texture combinations from sub-images of 3 scales and 3 orientations
      - DWT: discrete wavelet transformation using quadrature mirror filters
      - 4 sub-images + wavelets in (horizontal, vertical, diagonal)
    - spreadness (scatter) and elongation (shape) for each texture channel

Filter Name	Resolution	Representation
Masks	Coarse	Appearance of culture colors
Spread	***Coarse***	Spatial concentration of a color
Elongation	Coarse	Shape of a color
Histograms	Medium	Distribution of colors
Average	Medium	Similarity comparison within the same culture color
Variance	Fine	Similarity comparison within the same culture color

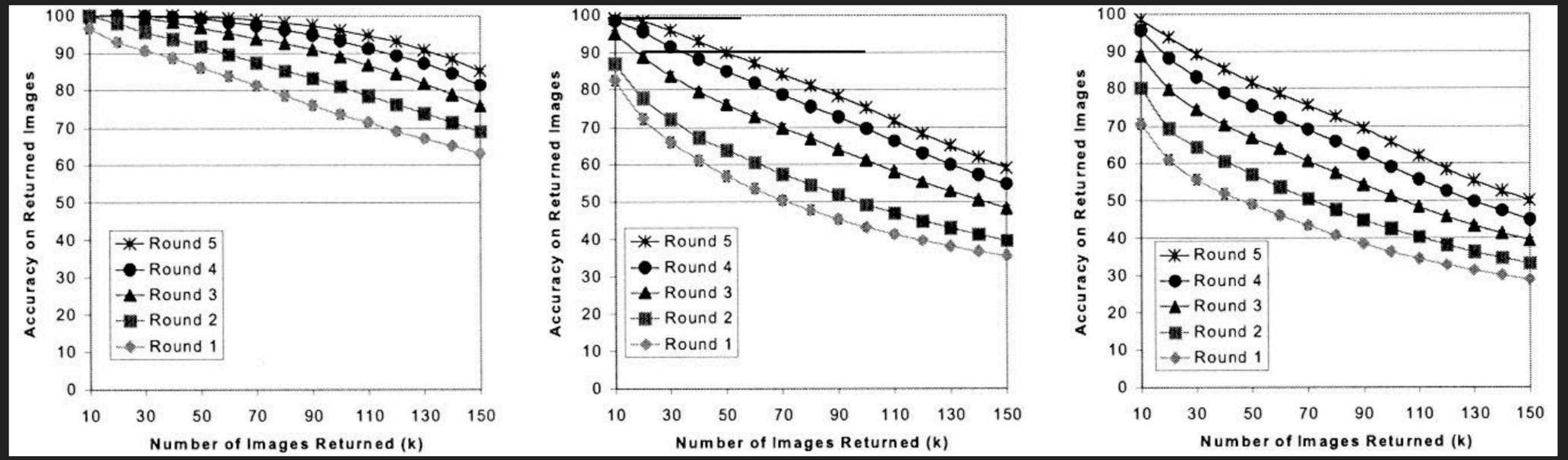


CONTENTS

DATASETS

- 4-category
  - Architecture, flowers, landscape, people
- 10-category
  - 4-category + bears, clouds, objectionable, tigers, tools, waves
- 15-category
  - 10-category + elephants, fabrics, fireworks, food, texture

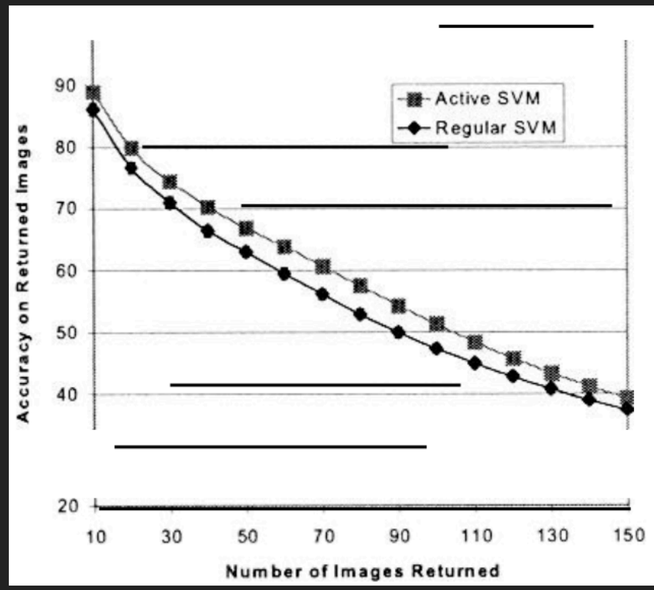
COMPARE WITH ITSELF



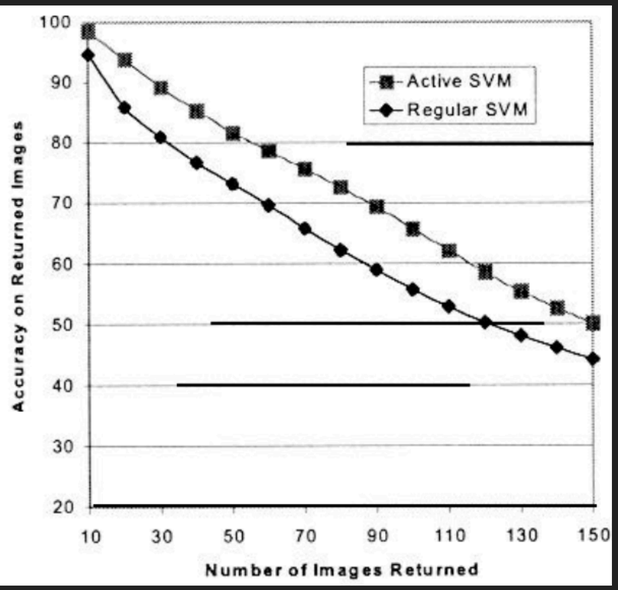
4-category 10-category 15-category

- Demonstrating SVMactive works as an active learning method
  - 4 rounds -> 100%, 95%, 88% accuracy on top-20 results
  - 5 rounds -> 99%, 84%, 76% accuracy on top-70 results

COMPARE WITH PASSIVE LEARNING BUT STILL USING SVM



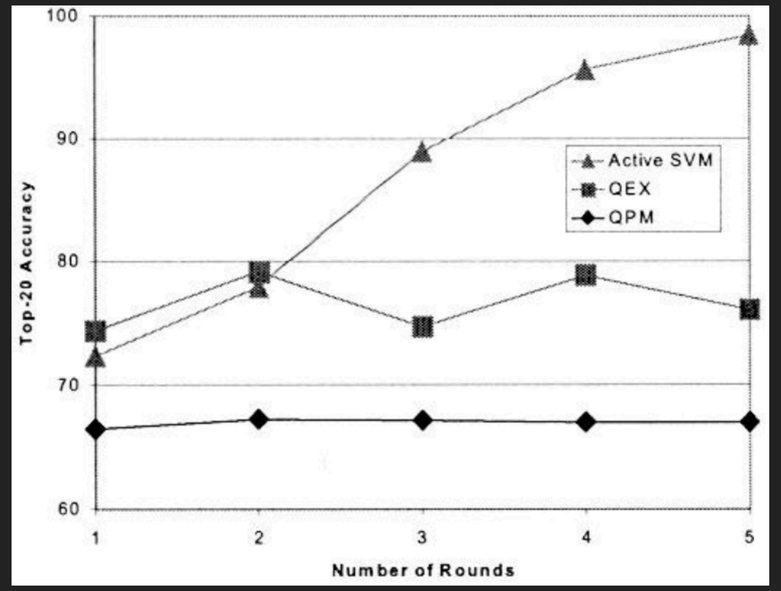
3 rounds on 15-category



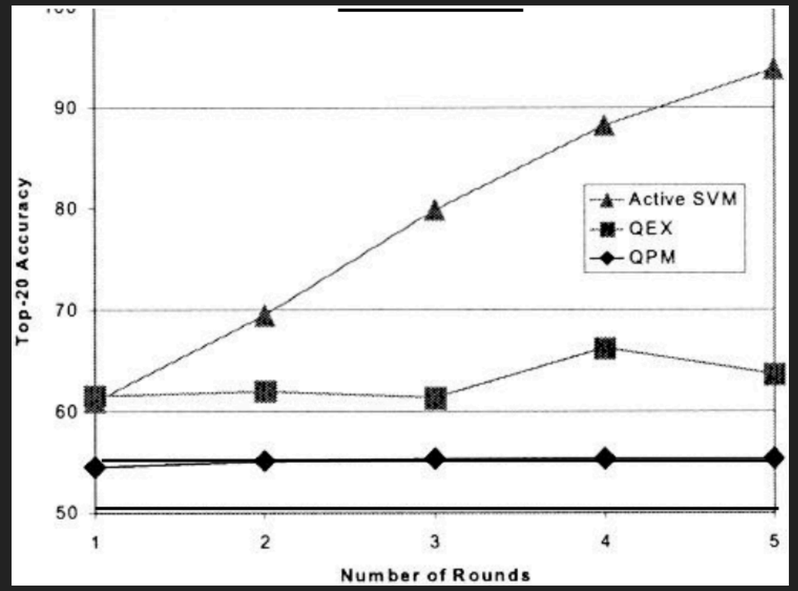
5 rounds on 15-category

- Demonstrating SVMactive works better than SVMs

COMPARE WITH OTHER SCHEMES



10-category

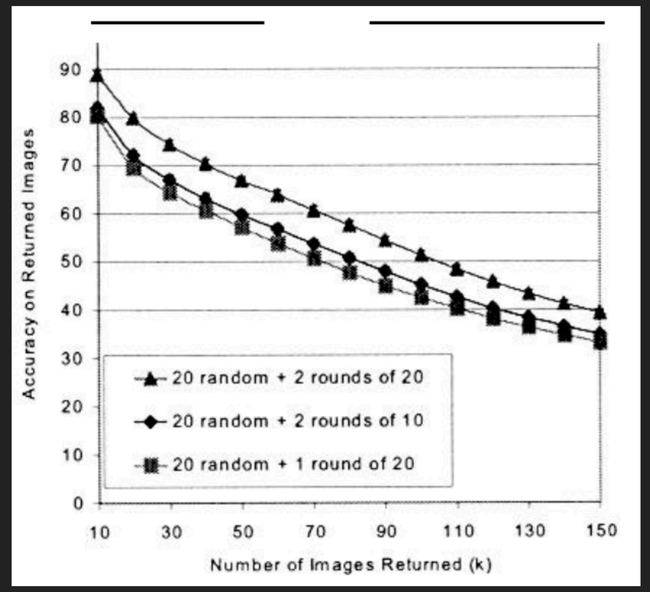


15-category

- Two traditional query refinement methods
  - QPM: query point movement, which use nearest-neighbor sampling of entire query
  - QEX: query expansion, which use neighbors of positive samples of previous round

- Demonstrating SVMactive outperforms others by significant margins
  - Traditional methods explore the space rather slow, while SVMactive is more aggressive

TUNING PARAMETERS



Tuning parameters on 15-category

- Two parameters
  - Number of images to be displayed in each round
  - Number of querying rounds

- The fewer images displayed per round, the lower the performance
- Recommend 20 images per round considering the negligible cost

OTHER ASPECTS

Texture features	Top-50 Accuracy
None	80.6 ± 2.3
Fine	85.9 ± 1.7
Medium	84.7 ± 1.6
Coarse	85.8 ± 1.3
Ail	86.3 ± 1.8

texture features

	Top-50	Top-100	Top-150
Degree 2 Polynomial	95.9 ± 0.4	86.1 ± 0.5	72.8 ± 0.4
Degree 4 Polynomial	92.7 ± 0.6	[illegible]	69.0 ± 0.5
Radial Basis	06.8 ± 0.3	89.1 ± 0.4	76.0 ± 0.4

kernel functions

Dataset	Dataset Size	round of 20 queries (secs)	Computing final SVM	Retrieving top 150 images
4 Cat	602	0.34 ± 0.00	0.5 ± 0.01	0.43 ± 0.02
10 cat	1277	[illegible] 0.01	1.03 ± d.03	0.93 ± 0.03
15 cat	1920	1.09 ± 0.02	1.74 [illegible]	1.37 ± 0.04

Avg run times in seconds

- runtime resource: 1x Sun Workstation

CONCLUSION

- SVMactive achieves
  - Consistently high accuracy on a wide variety of desired results
  - Both high efficiency and high precision when querying large amount of images (a few thousands, is large at that time of 2001)
  - Does not require an explicit semantical layer
- Possible directions of improvement
  - O(N) time complexity
  - Speed up based on clustering/indexing
  - Eliminating seed
  - Transduction

AUTHORS

Simon Tong · 3rd
Principal Engineer at Google
Mountain View, California, United States

Experience

Principal Engineer
Google
2001 - Present · 20 yrs 11 mos

- Co-designed one of the most broadly used machine learning systems at Google.
- Co-designed multiple systems that each increased revenue by a few to several percent.
- Designed and implemented systems that analyze billions to trillions of data instances.
- Applied machine learning and information retrieval techniques to many problems ranging from search ranking to ad targeting to hiring analysis.
- 76 US patents granted. Around 10 patents pending.

Education

Stanford University
Ph.D., Machine Learning, Computer Science
1997 - 2001

- Thesis title: "Active Learning: Theory and Applications".
- Citation count for three most popular papers: 3112, 1676, 239.
- Received 2020 ACM Multimedia "Test of time" honorable mention.
- First candidate to graduate in class for computer science.

University of Oxford
B.A., Mathematics and Computation
1994 - 1997

Activities and societies: St John's College.

- First Class Honours.
- Ranked joint first in year.

Edward Chang

Adjunct Professor, Computer Science, Stanford
AI/NLP Advisor, SmartNews

President, HTC Healthcare (DeepQ), 2012 - 2021
Director of Research, Google, 2006 - 2012
Professor, University of California, Santa Barbara, 1999 - 2006

Bio: Linkedin Profile
Contact: echang@cs.stanford.edu

Stanford OVAL: Open Virtual Assistant Lab
Google/DeepQ Open AI Platform

Edward Chang, a pioneer of data-driven deep learning and parallel machine learning algorithms, is currently serves as an adjunct professor at Stanford CS department and AI/NLP advisor at SmartNews (a Japan's unicorn). His 2010/11 data-driven deep learning patents filed at Google and sponsorship to Stanford ImageNet project contributed to the current AI revolution.

Education

Stanford University
PhD, Electrical Engineering
1995 - 1999

Databases, Machine Learning, Information Retrieval

Stanford University
M.S., Computer Science
1992 - 1994
Grade: 4.1

Operating Systems, Distributed Databases

UC Berkeley College of Engineering

Operations research, optimization, programming language

HAPPY THANKS GIVING

APPENDIX

- Radial basis function kernel (RBF):
  - Induces boundaries by placing weighted Gaussians upon key training instances
    $K(\mathbf{u}, \mathbf{v}) = (e^{-\gamma(\mathbf{u}-\mathbf{v})\cdot(\mathbf{u}-\mathbf{v})})$
  - Computes the similarity or how close they are to each other $K(X_1, X_2) = exp(-\frac{||X_1 - X_2||^2}{2\sigma^2})$
    - σ is the variance (hyper-parameter)
    - $||X_1 - X_2||$ is the Euclidean distance

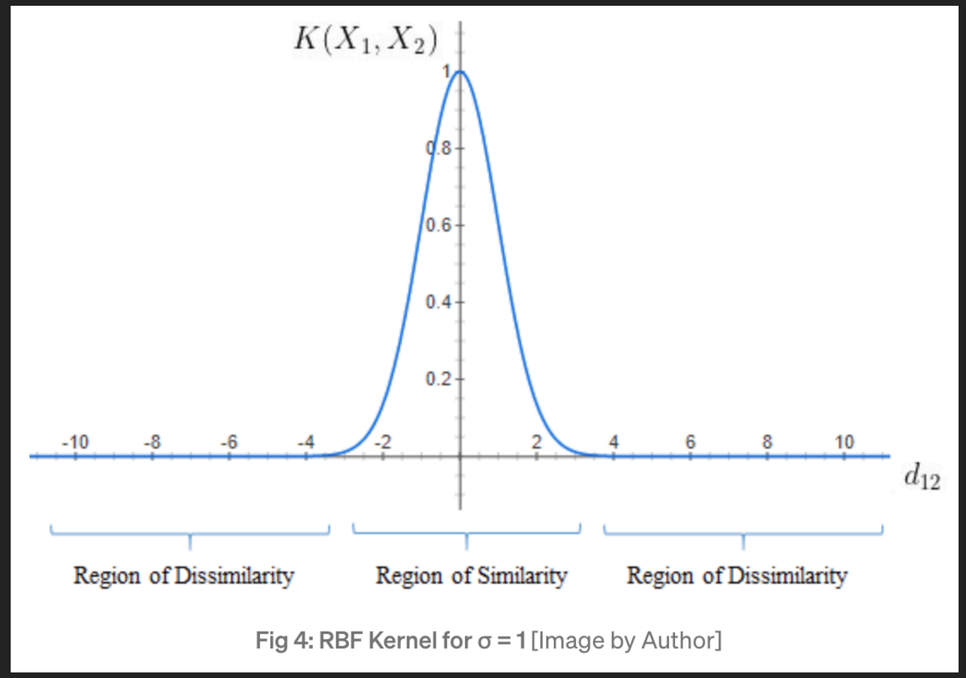


Fig 4: RBF Kernel for σ = 1 [Image by Author]

LINEARLY SEPARABLE FEATURE SPACE

- Very high dimension
- It's possible to modify any kernel so that the new induced feature space is linearly separable
  - Redefine all training instances: $\mathbf{x}_i\colon\ K(\mathbf{x}_i, \mathbf{x}_i) \leftarrow K(\mathbf{x}_i, \mathbf{x}_i) + \nu$
    - v is a positive regularization constant
    - Essentially achieve the same effect as the soft margin error function used in SVMs.