New Algorithms and Optimizations for Human-in-the-Loop Model Development

Doctoral Defense

Luciano Di Palma
Ecole Polytechnique, CEDAR group
luciano.di-palma@polytechnique.edu

Supervised by:
Yanlei Diao (advisor) and Anna Liu (co-advisor)

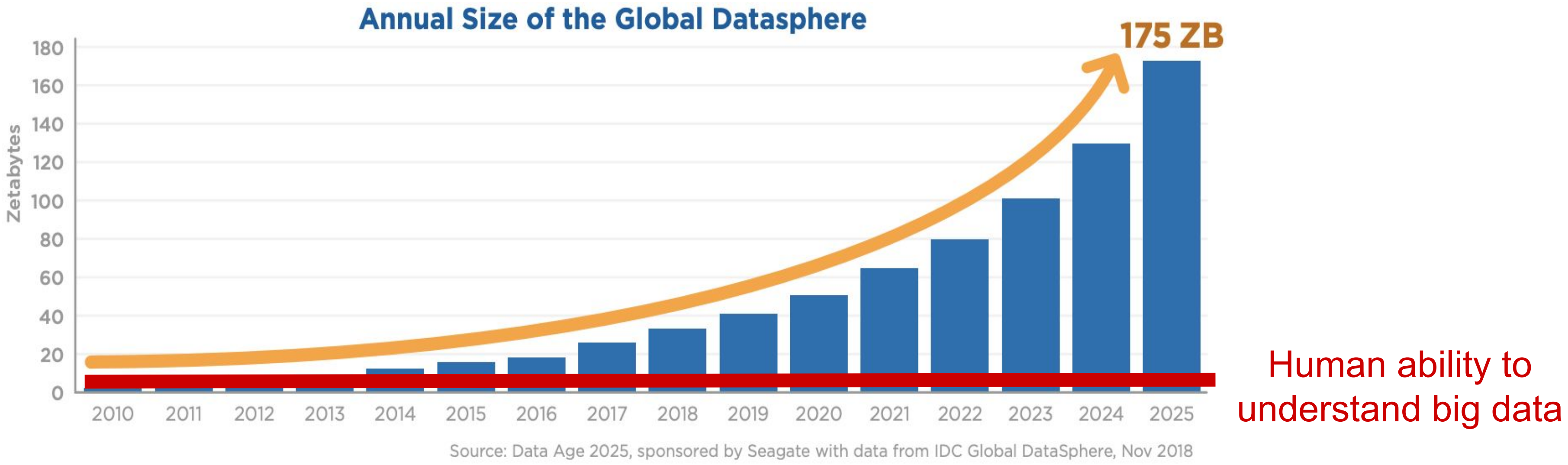
ÉCOLE POLYTECHNIQUE

INSTITUT POLYTECHNIQUE DE PARIS

Presentation Outline

1. **Introduction**
2. **Version Space Algorithms**
3. **A Factorized Version Space Algorithm**
4. **Learning a Factorized Classifier**
5. **Related Work**
6. **Summary**

The "Big Data" Era



There is an ever-increasing gap between the amount of available information and the human ability to derive high-value content from it

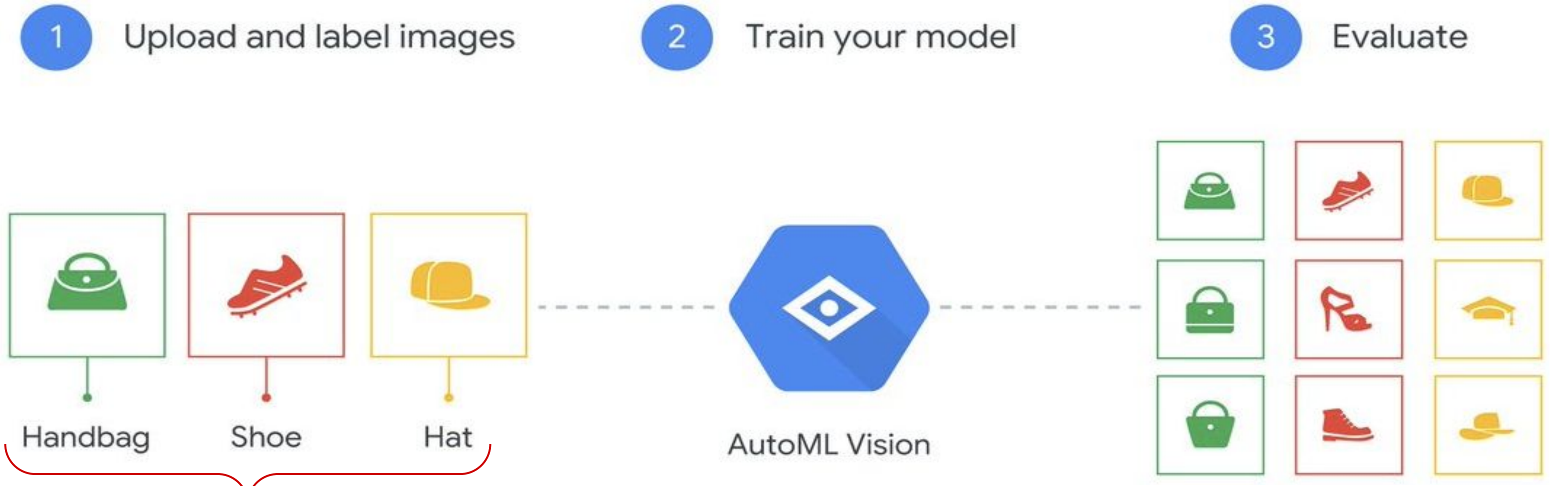
Challenge #1: Database Exploration

- Data is often stored in a database system
- **Database Exploration**: User cannot translate his / her interest into a database query
  - Must resort to a <u>manual exploration</u> process

$$
\begin{array}{llll}
\textbf{SELECT} * \textbf{FROM } \text{Cars } \textbf{WHERE} & \text{price} < 30000 & & \\
\textbf{SELECT} * \textbf{FROM } \text{Cars } \textbf{WHERE} & 20000 < \text{price} < 30000 & \textbf{AND } \text{body\_type} = \text{'sedan'} & \\
\textbf{SELECT} * \textbf{FROM } \text{Cars } \textbf{WHERE} & 20000 < \text{price} < 28000 & \textbf{AND } \text{body\_type} = \text{'sedan'} & \\
\textbf{SELECT} * \textbf{FROM } \text{Cars } \textbf{WHERE} & 20000 < \text{price} < 25000 & \textbf{AND } \text{body\_type} = \text{'sedan'} & \\
\textbf{SELECT} * \textbf{FROM } \text{Cars } \textbf{WHERE} & 20000 < \text{price} < 25000 & \textbf{AND } \text{body\_type} = \text{'sedan'} & \textbf{AND } \text{year} = 2018 \\
\textbf{SELECT} * \textbf{FROM } \text{Cars } \textbf{WHERE} & 20000 < \text{price} < 28000 & \textbf{AND } \text{body\_type} = \text{'sedan'} & \textbf{AND } \text{year} = 2018
\end{array}
$$

Database users lack automated tools for efficient data exploration

Challenge #2: Data Annotation



Where to obtain annotated data?

<u>Existing solutions</u>: crowdsourcing, manual labeling by IT teams, etc

"ML for everyone" needs tools for <u>accurate</u> and <u>automatic</u> annotation of large datasets

Human-in-the-Loop Model Development

Objective: Build an accurate classifier of the user interest from his / her feedback

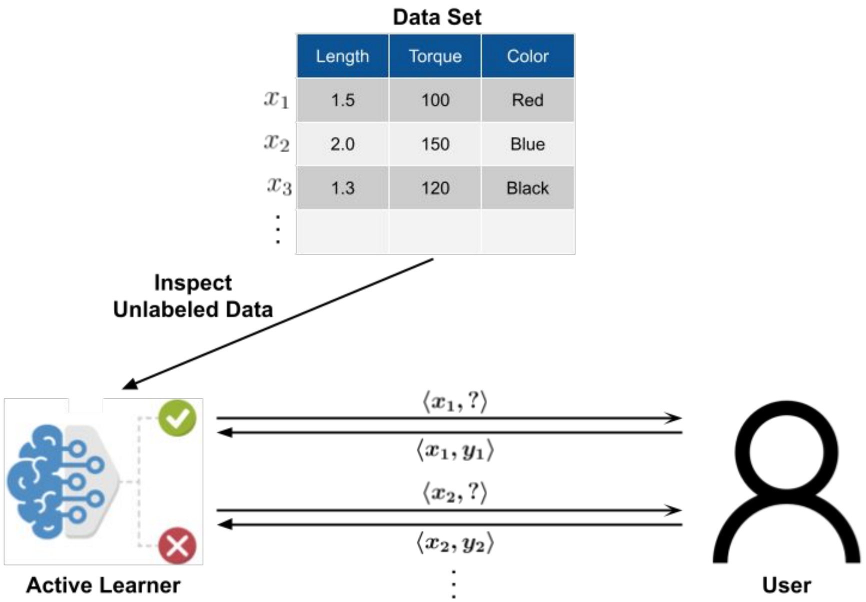
Challenge	**User Feedback**	**Final Classifier**
DB users need automated data exploration tools	User annotates a subset of points as “interesting” or “not interesting”	Build a classification model predicting objects of interest
ML platforms need efficient data annotation methods	User annotates only a well-chosen selection of data points	Train a classifier than can accurately label the entire data

HiL can effectively deal with both data exploration and annotation challenges!

Active Learning

Challenge: How to minimize the user annotation effort?

Active Learning: ML methods for training accurate classifiers with minimal labeled data

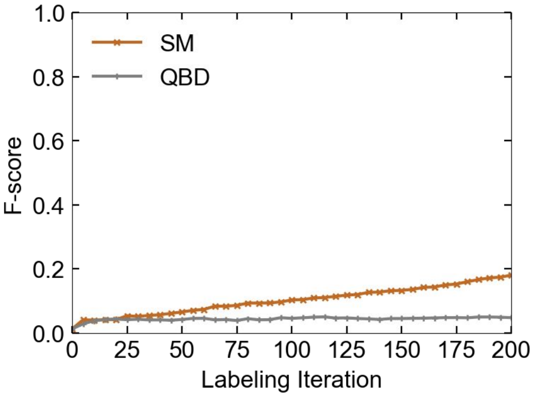


	Length	Torque	Color
x_1	1.5	100	Red
x_2	2.0	150	Blue
x_3	1.3	120	Black
$\vdots$			

AL needs fewer labeled examples than traditional ML techniques

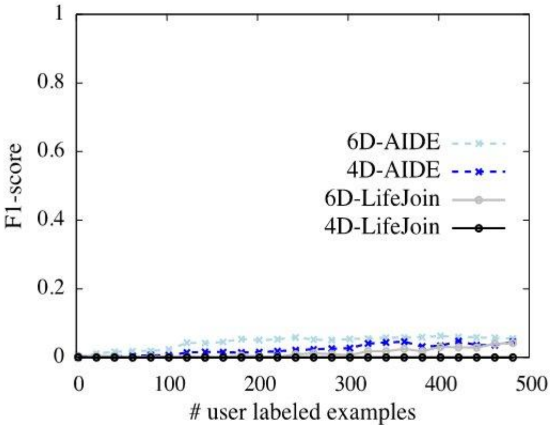
Active Learning over Large Datasets

Active Learning Techniques



6D model, 0.01% selectivity

Data Exploration Systems



4D and 6D models, 0.01% selectivity

Dataset: Sloan Digital Sky Survey (SDSS), 1% of PhotoObjAll table, 1.8M tuples

AIDE: K. Dimitriadou, O. Papaemmanouil, and Y. Diao. “Aide: an active learning approach for interactive data exploration”. Transactions on Knowledge and Data Engineering, 2016.
LifeJoin: A. Cheung, A. Solar-Lezama, and S. Madden. “Using program synthesis for social recommendations.” Conference on Information and Knowledge Management (CIKM), 2012.
Simple Margin (SM): S. Tong and D. Koller. Support Vector Machine active learning with applications to text classification. *Journal of Machine Learning Research*, 2:45–66, 2001.
Query-by-Disagreement (QBD): B. Settles. *Active Learning*. Morgan & Claypool Publishers, 2012.

Our Contributions

Objective: Design new AL techniques that:

- ➔ Overcome the slow start problem
- ➔ Provide interactive performance for user labeling

Contributions:

- ➔ *Version Space Algorithms*: AL algorithm with strong theoretical guarantees on performance and optimizations for interactive performance
- ➔ *Factorization*: Leverage additional information from the user to expedite convergence
- ➔ *Factorized Classifiers*: A new classification model mimicking the human decision-making

Presentation Outline

Overview of Active Learning

Notation

- **Dataset**: $\mathcal{X} = \{x_1, \dots, x_N\}$
- **User Label**: $y_i \in \{\pm 1\}$
- **Hypothesis Set**: set of classifiers $h \in \mathcal{H}$
- **Labeled Set**: $(x, y) \in \mathcal{L}$
- **Unlabeled Set**: $\mathcal{U} = \{x \in \mathcal{X} : (x, \pm 1) \notin \mathcal{L}\}$

Goal: find $h^* \in \mathcal{H}$ matching the user interest with minimal annotation effort

$$h^*(x_i) = y_i, \ \forall\, i$$

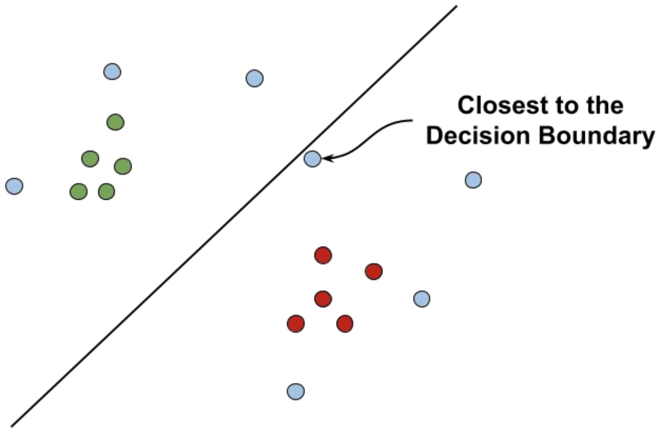
$$|\mathcal{L}| \ll |\mathcal{X}|$$

Active Learning Strategies

Main Idea: incrementally build $\mathcal{L}$ by selecting the <u>most informative</u> points to label

Uncertainty Sampling

- ➔ Information = “Uncertainty”
- ➔ Slow-converging, but efficient



And several other approaches…

- ➔ Entropy minimization
- ➔ Expected error reduction
- ➔ Information-theory based

B. Settles. *Active Learning*. Morgan & Claypool Publishers, 2012

Version Space Algorithms

Why Version Space Algorithms?

- ➔ They provide <u>strong theoretical guarantees</u> on convergence speed

What is the Version Space?

- ➔ Set of all classifiers consistent with the labeled data

$$\mathcal{V} = \{h \in \mathcal{H} : h(x) = y \text{ for all } (x, y) \in \mathcal{L}\}$$

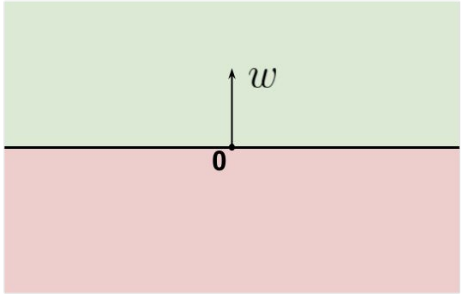
Properties of the Version Space

- ➔ The optimal classifier h^* is always inside $\mathcal{V}$ (assuming no labeling mistakes)
- ➔ It shrinks as more data is labeled

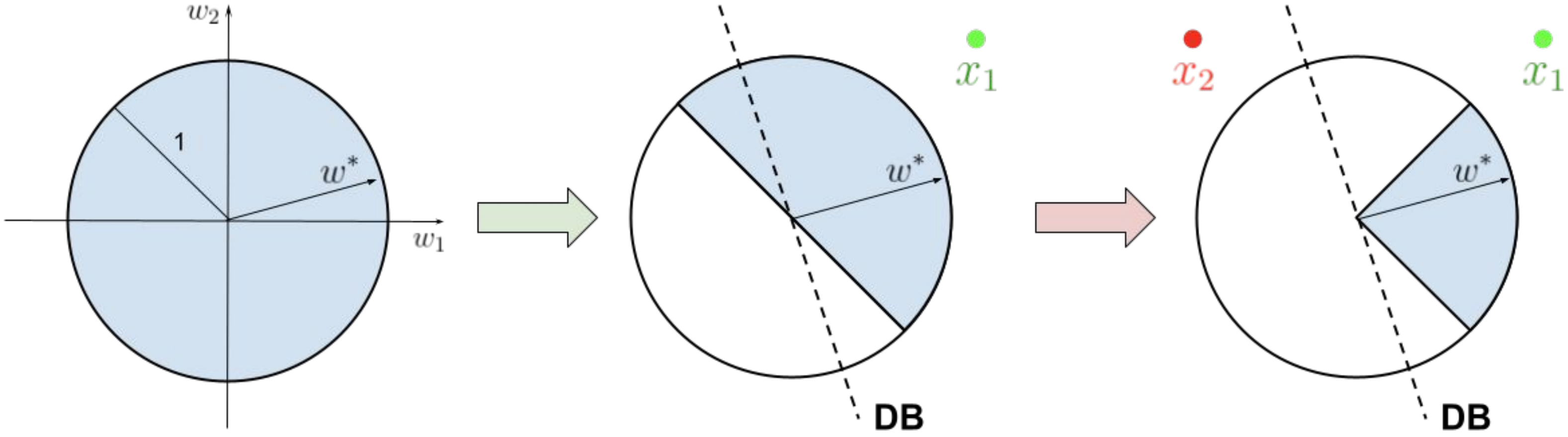
Objective: Reduce $\mathcal{V}$ as quickly as possible

Example: Homogeneous Linear Classifiers (2D)

$$h_w(x) = sign(x^T w), \text{ for } \|w\| \leq 1$$

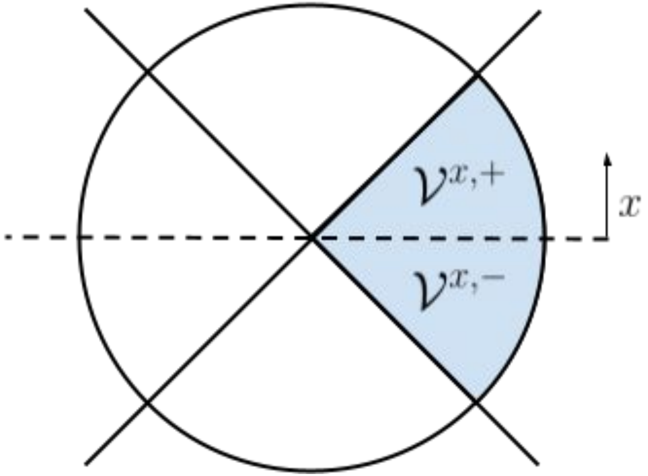


$$\mathcal{V} = \{w \in \mathbb{R}^2 : yx^T w > 0 \text{ for } (x, y) \in \mathcal{L} \text{ and } \|w\| \leq 1\}$$



Generalized Binary Search (GBS)

Bisection Rule: choose the data point splitting the version space in half



➔ **Cut probabilities**: $p_{x,\pm} = \dfrac{vol(\mathcal{V}^{x,\pm})}{vol(\mathcal{V})}$

➔ **Selection strategy**: $x_* \in \operatorname*{argmin}_{x \in \mathcal{U}} |p_{x,+} - 0.5|$

S. Dasgupta. Analysis of a greedy active learning strategy. In *Advances in Neural Information Processing Systems 17*, pages 337–344, 2005.
K. Trapeznikov, V. Saligrama, and D. Castañón. Active boosted learning (ActBoost). In *International Conference on Artificial Intelligence and Statistics*, volume 14, 2011.

Theoretical Guarantees (GBS)

- ➔ $\mathcal{A}$: an Active Learning algorithm
- ➔ h: the classifier $h \in \mathcal{H}$ matching the user preference
- ➔ $cost(\mathcal{A}, h)$: # of queries that $\mathcal{A}$ takes to identify h
- ➔ $cost(\mathcal{A})$: average cost of $\mathcal{A}$ across all possible labelings h

Let $OPT = \min_{\mathcal{A}} cost(\mathcal{A})$. Then, the GBS strategy satisfies:

$$cost(GBS) \leq OPT \cdot \left(1 + \ln \frac{1}{\min_h \pi([h])}\right)^2$$

S. Dasgupta. Analysis of a greedy active learning strategy. In *Advances in Neural Information Processing Systems 17*, pages 337–344, 2005.
D. Golovin and A. Krause. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *Journal of Artificial Intelligence Research*, 42:427–486, 2011

Limitations of VS Algorithms

- ➔ The GBS is **too expensive** to run in practice
- ➔ In the literature, several approximations have been introduced:

	Slow Convergence	Fast Convergence
Low Time Cost	**SM**, **QBD**	**OptVS**
High Time Cost		**ALuMA**, **KQBC**

We propose **OptVS**, an optimized VS algorithm offering:

- ➢ Fast convergence speed
- ➢ Low running time
- ➢ Theoretical guarantees

Simple Margin (SM): S. Tong and D. Koller. Support Vector Machine active learning with applications to text classification. *Journal of Machine Learning Research*, 2:45–66, 2001.
Kernel Query-by-Committee (KQBC): R. Gilad-Bachrach, A. Navot, and N. Tishby. Query-by-committee made real. In *Advances in Neural Information Processing Systems 18,* 2006.
Query-by-Disagreement (QBD): B. Settles. *Active Learning*. Morgan & Claypool Publishers, 2012.
ALuMA: A. Gonen, S. Sabato, and S. Shalev-Shwartz. Efficient active learning of halfspaces: an aggressive approach. *Journal of Machine Learning Research*, 14(1): 2583–2615, 2013

OptVS: Theoretical Formulation

Objective: Efficiently realize the GBS algorithm over kernel classifiers

Kernel Classifiers: Given a kernel k and its feature map $\phi : \mathbb{R}^d \rightarrow \mathcal{F}$, we define:

$$h_f(x) = sign\langle\phi(x), f\rangle_{\mathcal{F}}$$

$$\mathcal{V} = \{f \in \mathcal{F} : ||f||_{\mathcal{F}} \leq 1 \text{ and } y\langle\phi(x), f\rangle_{\mathcal{F}} > 0 \text{ for all } (x, y) \in \mathcal{L}\}$$

Problem: Parameter space is very high-dimensional, possibly infinite!

➔ Estimating the version space size is intractable.

Dimensionality Reduction

- ➔ $\mathcal{X} = \{x_1, \ldots, x_N\}$: dataset
- ➔ $\mathcal{L} = \{(x_i, y_i)\}_{i=1}^{t}$: user-labeled points
- ➔ $K_i^j = k(x_i, x_j)$: $N \times N$ kernel matrix
- ➔ L_i^j : Cholesky decomposition of K

Our theoretical result: The cut probabilities for kernel classifiers satisfy:

$$p_{x_j,\pm} = \mathbb{P}(\pm w^T \ell_j > 0)$$

Reduced Version Space

$$w \sim Unif(W_t)$$

$$W_t = \{w \in \mathbb{R}^{t+1} : \|w\| \leq 1 \text{ and } y_i w^T \ell_i \geq 0,\ 1 \leq i \leq t\}$$

Partial Cholesky Factor

$$\ell_i = \left(L_i^1, \ldots, L_i^t, \sqrt{K_i^i - \sum_{k=1}^{t} (L_i^k)^2} \right)$$

Computation reduced to the <u>linear case</u>, scaling with the <u>number of labeled points</u>

Computing the Cut Probabilities

With the dimensionality controlled, how do we <u>estimate the cut probabilities</u>?

$$p_{x_j,\pm} = \boxed{\mathbb{P}}(\pm \boxed{w}^T \boxed{\ell_j} > 0)$$

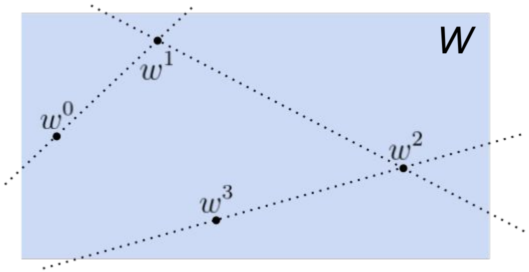
Sampling estimation via "Law of Large Numbers"

Hit-and-Run + Rounding for efficient sample generation

Simple matrix manipulations

$$W_t = \{w \in \mathbb{R}^{t+1} : \|w\| \leq 1 \text{ and } y_i w^T \ell_i \geq 0,\ 1 \leq i \leq t\}$$

Probability Estimation and Optimizations



Hit-and-Run

- ➔ SOTA algorithm for sampling uniformly over convex bodies
- ➔ Generates a Markov Chain inside *W*
- ➔ Efficient sample generation

Probability Estimation

- ➔ Law of Large Numbers for Markov Chains
- ➔ Draws samples from a single chain (fast!)

$$p_{x_j,+} = \lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}\left(\ell_j^T w^i > 0\right)$$

Rounding

- ➔ Preprocessing step of Hit-and-Run for improved mixing time
- ➔ **Caching**: re-use previous transformations to warm-start computation
- ➔ **Numerical Stability**: adapt the algorithm to the particular shape of the VS

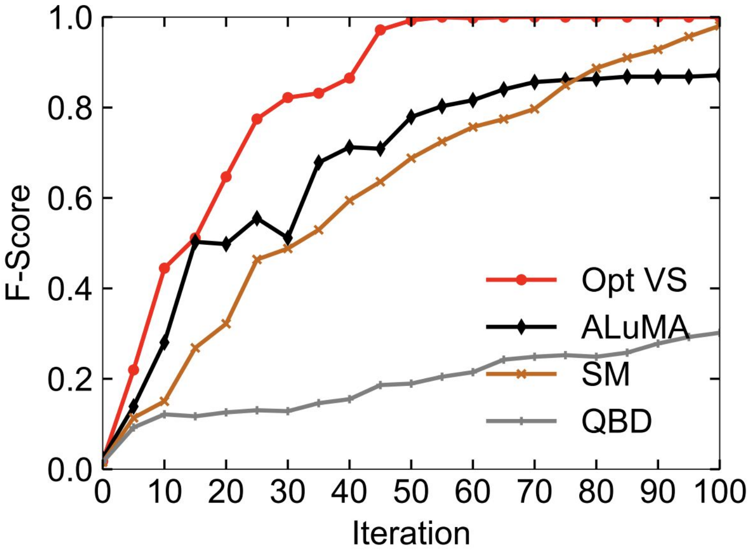
Experimental Evaluation

- **Datasets**
  - Sloan Digital Sky Survey (SDSS)
    - PhotoObjAll table, 190 million tuples
    - 1% sample pool, 1.8 million tuples, 4.9GB
    - 11 user interest patterns from the SDSS query release
  - Car database
    - Extracted from teoalida.com
    - 5,622 tuples
    - 18 user interest patterns from a user study

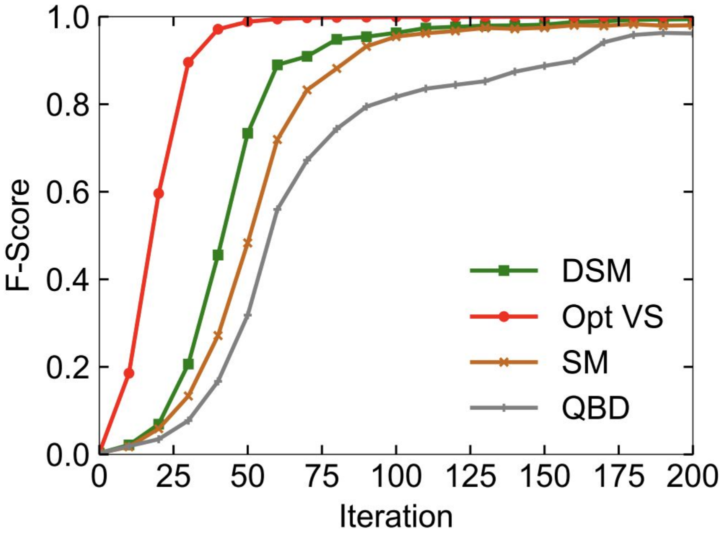
- **Algorithms**
  - VS Algorithms: **Simple Margin (SM)**, **Query-by-Disagreement (QBD)**, **ALuMA**
  - Data Exploration: **Dual Space Model (DSM)**

Simple Margin (SM): S. Tong and D. Koller. Support Vector Machine active learning with applications to text classification. *Journal of Machine Learning Research*, 2:45–66, 2001.
Query-by-Disagreement (QBD): B. Settles. *Active Learning*. Morgan & Claypool Publishers, 2012.
ALuMA: A. Gonen, S. Sabato, and S. Shalev-Shwartz. Efficient active learning of halfspaces: an aggressive approach. *Journal of Machine Learning Research*, 14(1): 2583–2615, 2013
Dual-Space Model (DSM): E. Huang, L. Peng, L. D. Palma, A. Abdelkafi, A. Liu, and Y. Diao. Optimization for active learning-based interactive database exploration. *Proceedings of the VLDB Endowment*, 12(1):71–84, 2018.

Evaluating OptVS (Performance)



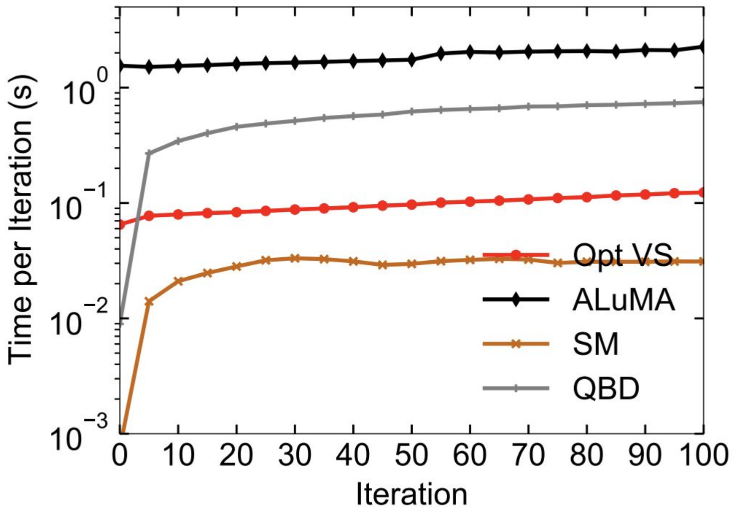
F-score for Car 10 (0.36%, 8D, 31D enc)



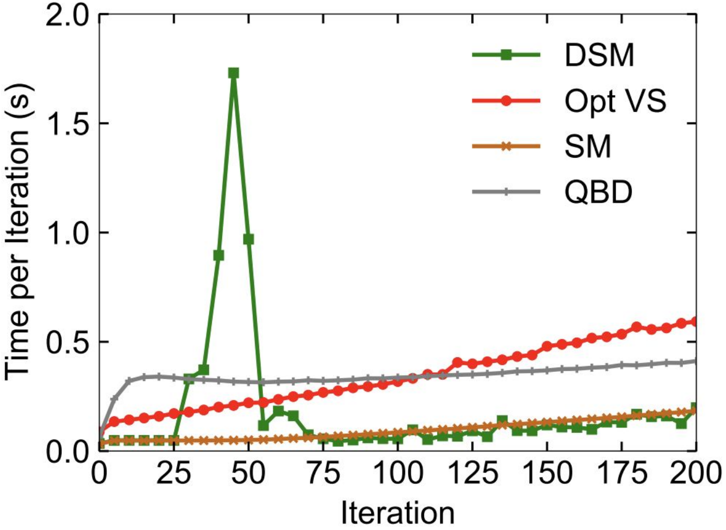
F-score for SDSS 02 (0.1%, 2D)

OptVS outperforms state-of-the-art VS algorithms and unfactorized DSM

Evaluating OptVS (Efficiency)



Time for Car 05 (0.23%, 6D, 418D enc)



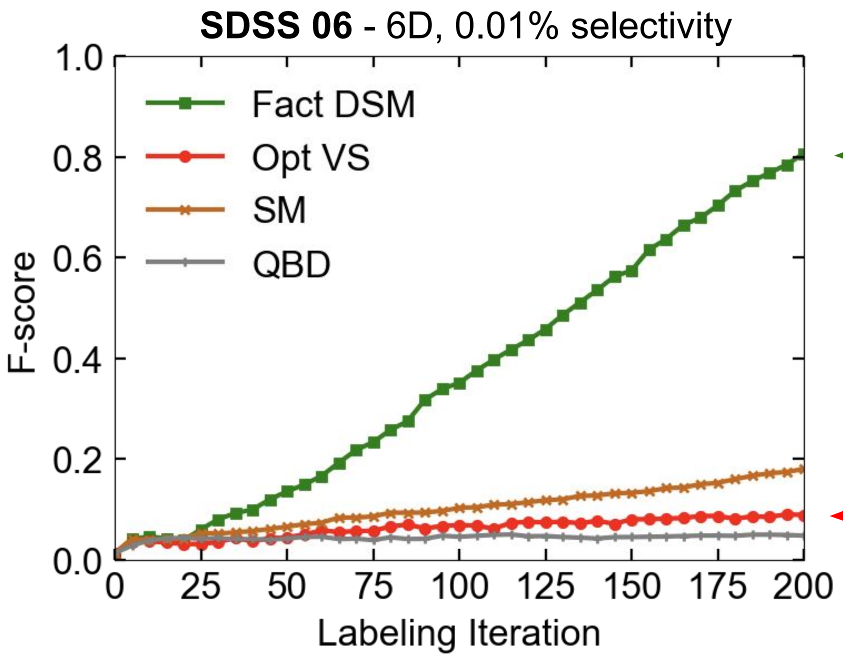
Time for SDSS 02 (0.1%, 2D)

OptVS runs under interactive performance at all times

Presentation Outline

1. Introduction
2. Version Space Algorithms
3. **A Factorized Version Space Algorithm**
4. Learning a Factorized Classifier
5. Related Work
6. Summary

Limitations of Active Learning



DSM can improve upon AL by leveraging a **factorization** from the user

In practice, our VS technique can still suffer from slow convergence!

Can we extend our VS techniques to leverage the factorization information?

What is Factorization?

Idea: Leverage additional insights from the user labeling process

Example: A customer looking for cars of interest may have several concerns:

Q1: Is the gas mileage good enough?

Q2: Is the vehicle spacious enough?

Q3: Is the color a preferred one?

The global decision (interesting or not) is factorized into simple yes / no questions

We wish to leverage:
- ➔ Which **attributes** are involved in each question
- ➔ The **user feedback** (yes / no) for each question

Factorized Version Space: Intuition

Version Space V (size = 16)

Color (B/R)	Size (S/L)	Label
B(lack)	L(arge)	{-, +}
B(lack)	S(mall)	{-, +}
R(ed)	L(arge)	{-, +}
R(ed)	S(mall)	{-, +}

A labeled example

(BL, '–')

Version Space V (size = 8)

Color (B/R)	Size (S/L)	Label
B(lack)	L(arge)	{ - }
B(lack)	S(mall)	{-, +}
R(ed)	L(arge)	{-, +}
R(ed)	S(mall)	{-, +}

(a) Without factorization

Factorization Structure
{Color}, {Size}

Version Space V (size = 16)

Color (B/R)	Label
B(lack)	{-, +}
R(ed)	{-, +}

Size (S/L)	Label
L(arge)	{-, +}
S(mall)	{-, +}

A labeled example

((B, '–'), (L, '+'))

Version Space V (size = 4)

Color (B/R)	Label
B(lack)	{ - }
R(ed)	{-, +}

Size (S/L)	Label
L(arge)	{ + }
S(mall)	{-, +}

(b) With factorization

Factorization leads to a faster reduction of the VS!

Factorized Version Space: Formalism

Given a factorization $(A^1, \ldots, A^S)$, we define:

Factorized Hypothesis Set: we model one classifier per subspace

$$\mathcal{H}_f = \mathcal{H}^1 \times \ldots \times \mathcal{H}^S$$

where each tuple $H = (h^1, \ldots, h^S)$ is viewed as a multi-label classifier

$$H(x) = (h^1(x^1), \ldots, h^S(x^S)) \in \{-, +\}^S$$

Factorized Version Space: applying the usual definition, we have

$$\mathcal{V}_f = \mathcal{V}^1 \times \ldots \times \mathcal{V}^S$$

where the *version subspaces* are defined as

$$\mathcal{V}^s = \{h \in \mathcal{H}^s : h(x^s) = y^s \text{ for all } (x, y) \in \mathcal{L}\}$$

Factorized Bisection Rule

By applying the VS bisection rule over $\mathcal{V}_f$, we obtain:

$$\arg\max_{x \in \mathcal{U}} 1 - \sum_{y \in \{-,+\}^S} p_{x,y}^2, \quad \text{where} \quad p_{x,y} = \frac{vol(\mathcal{V}_f^{x,y})}{vol(\mathcal{V}_f)}$$

In particular, we can show that the above expression is equivalent to:

$$\arg\max_{x \in \mathcal{U}} 1 - \prod_{s=1}^{S} (1 - 2p_{x,+}^s p_{x,-}^s), \quad \text{where} \quad p_{x,\pm}^s = \frac{vol(\mathcal{V}_s^{x^s,\pm})}{vol(\mathcal{V}_s)}$$

Thus, we only need to repeat the usual VS computations for each subspace!

Theoretical Guarantees

- ➔ $\mathcal{A}_f$: an Active Learning algorithm making use of the partial labels information
- ➔ H: the classifier $H \in \mathcal{H}_f$ matching the user preference in each subspace
- ➔ $cost(\mathcal{A}_f, H)$: # of queries that $\mathcal{A}_f$ takes to identify H in <u>every</u> subspaces
- ➔ $cost(\mathcal{A}_f)$: average cost of $\mathcal{A}_f$ across all possible labelings H

Let $OPT_f = \min_{\mathcal{A}_f} cost(\mathcal{A}_f)$. Then, our Factorized VS strategy satisfies:

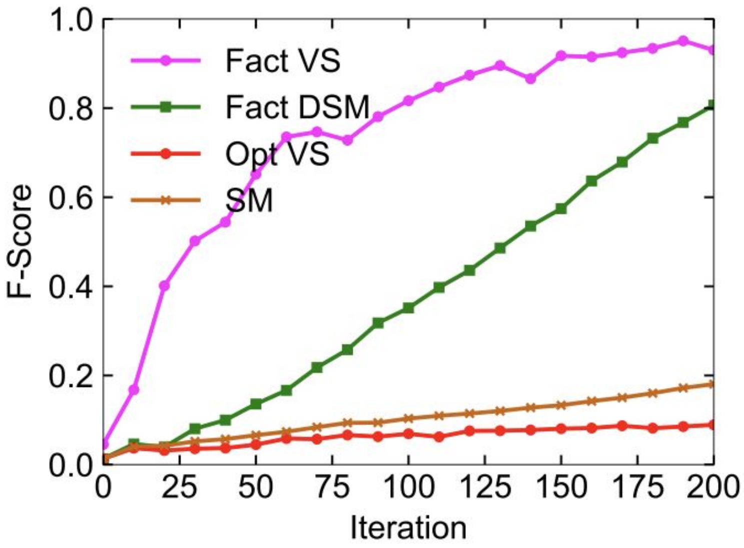
$$cost(FactVS) \leq OPT_f \left(1 + \sum_{s=1}^{S} \log \frac{1}{\min_{h^s} \pi^s([h^s])}\right)^2$$

S. Dasgupta. Analysis of a greedy active learning strategy. In *Advances in Neural Information Processing Systems 17*, pages 337–344, 2005.
D. Golovin and A. Krause. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *Journal of Artificial Intelligence Research*, 42:427–486, 2011

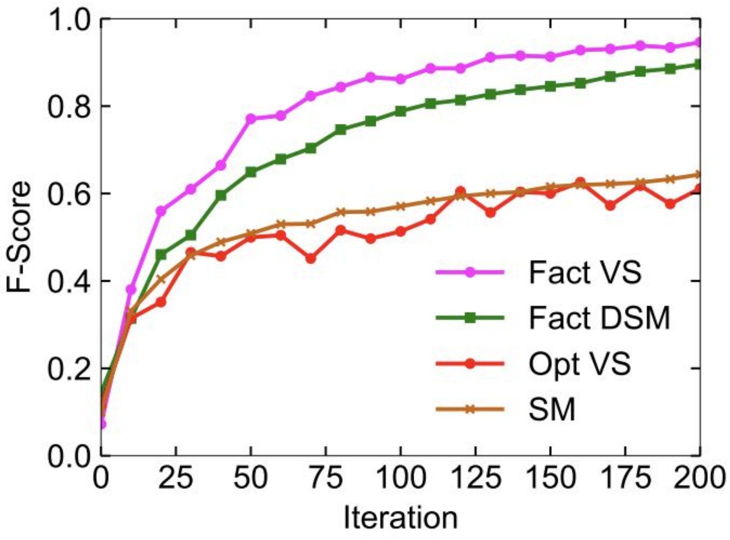
SDSS Results

SDSS 06: $(rowc - 682.5)^2 + (colc - 1022.5)^2 < 280^2$ **AND** $(150 < ra < 240$ AND $40 < dec < 70)$ **AND** $rowv^2 + colv^2 > 0.2^2$

SDSS 11: $(u - g > 2.0$ OR $u > 22.3)$ **AND** $0 \leq i \leq 19$ **AND** $g - r > 1$ **AND** $r - i < 0.08 + 0.42 * (g - r - 0.96)$ OR $g - r > 2.26$ **AND** $i - z < 0.25$



SDSS 06 (0.01%, 6D)



SDSS 11 (0.1%, 5D)

FactVS outperforms both DSM and AL algorithms in high-dimensional exploration!

Presentation Outline

1. Introduction
2. Version Space Algorithms
3. A Factorized Version Space Algorithm
4. **Learning a Factorized Classifier**
5. Related Work
6. Summary

Questions that remain...

What if the user does not know the factorization?

- ➔ Lack understanding of own decision-making process
- ➔ Unfamiliarity with the data distribution

Can we learn it from the labeled data?

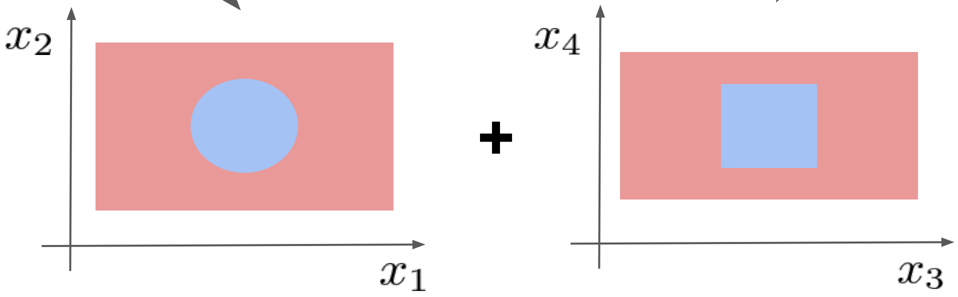
$$\boxed{(x_1 - 682.5)^2 + (x_2 - 1022.5)^2 < 90^2} \text{ AND } \boxed{180 < x_3 < 210} \text{ AND } \boxed{50 < x_4 < 60}$$

$$\{x_1, x_2\}, \{x_3\}, \{x_4\}$$

Questions that remain...

Can we learn a classifier that mimics the user decision-making process?

$$\boxed{(x_1 - 682.5)^2 + (x_2 - 1022.5)^2 < 90^2} \text{ AND } \boxed{180 < x_3 < 210 \text{ AND } 50 < x_4 < 60}$$



= Final Prediction

Advantages: accuracy and interpretability

Learning a Factorized Classifier

Intuition: The final prediction is the combination of simple, independent decisions

$$Y = \boxed{F}(\underbrace{Y^1, \ldots, Y^{\boxed{S}}})$$

Decision Function
$F : \{0,1\}^S \rightarrow \{0,1\}$

Subspatial Classification Models
Each label $Y^s \in \{0,1\}$ depends only on a subset of attributes A^s

Number of Subspaces
$S \in \mathbb{N}$

Challenge: Several interconnected components to learn

Conjunctive Assumption

Conjunctivity: Model the user decision-making as a list of requirements

$$F(y^1, \ldots, y^S) = y^1 \wedge \ldots \wedge y^S$$

Not very restrictive since:

- ➔ Every boolean function can be written in Conjunctive Normal Form (CNF)
- ➔ Backed up by our user study

Linearity Assumption

Linearity: Subspace labels can be accurately modeled by logistic regressors

$$\mathbb{P}(Y^s = 1 | X = x) = \sigma(b^s + \langle x, w^s \rangle)$$

Advantages:

- ➔ Support for complex user interest patterns
- ➔ Easy estimation of factorization structure $\longrightarrow$ $\boldsymbol{A}^s = \{i \in \boldsymbol{A} : w_i^s \neq 0\}$
- ➔ Efficient optimization (parameterized model)

The Factorized Linear Model (FLM)

Factorized Linear Model: By putting the previous points together, we define:

$$\mathbb{P}(Y = 1 | X = x) = p_{b,W}(x) = \prod_{s=1}^{S} \sigma(b^s + \langle x, w^s \rangle)$$

Combination of independent, linear classifiers!

Learning Algorithm: Simple minimization of the cross-entropy loss function:

$$\min_{b \in \mathbb{R}^d, W \in \mathbb{R}^{S \times d}} \frac{1}{m} \sum_{i=1}^{m} -y_i \log p_{b,W}(x_i) - (1 - y_i) \log(1 - p_{b,W}(x_i))$$

Non-convex, differentiable optimization problem

Feature and Subspace Selection

Objective: Improve <u>interpretability</u> by limiting the number of relevant features per subspace

➔ Irrelevant features = zero weights

➔ **Idea**: Induce sparsity to the weights by adding *penalty terms*

$$\min_{b,W} \boxed{\frac{1}{m}\sum_{i=1}^{m} -y_i \log p_{b,W}(x_i) - (1-y_i)\log(1-p_{b,W}(x_i))} + \boxed{\sum_{s=1}^{S} \|w^s\|_1} + \boxed{\sum_{s=1}^{S} \|w^s\|_2}$$

Classification Error
Induces an accurate solution

Lasso Penalty
Removes irrelevant features in each subspace

Group Lasso Penalty
Removes irrelevant subspaces

Automatic selection of most relevant features and number of subspaces

Practical Example - SDSS Query 09

Weight Matrix

u	g	r	i	z
0.002	0.002	-0.001	0.002	-0.002
-0.003	0.003	0.002	-0.003	-0.003
-0.4	-50	46	0.1	-0.1
-0.1	-0.04	0.1	42	-43
-0.001	0.002	0.001	0.002	0.002
0.003	0.001	0.001	0.001	-0.001
0.002	0.003	0.002	-0.001	0.002
-0.002	0.002	0.001	-0.002	0.002
-35	38	0.02	0.01	-0.2
-0.1	0.1	35	-33	-1

Pruning

Pruned Weight Matrix

u	g	r	i	z
-0.4	-50	46	0.1	-0.1
-0.1	-0.04	0.1	42	-43
-35	38	0.02	0.01	-0.2
-0.1	0.1	35	-33	-1

Feature Selection

Selection Matrix

u	g	r	i	z
0	1	1	0	0
0	0	0	1	1
1	1	0	0	0
0	0	1	1	0

Attributes: *u, g, r, i, z*

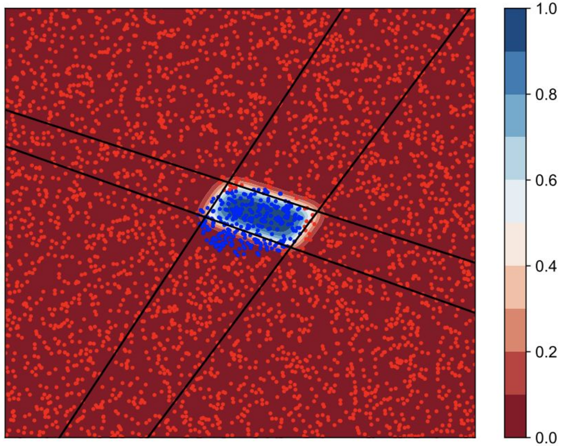
Query predicate: $u - g < 0.4 \textbf{ AND } g - r < 0.7 \textbf{ AND } r - i > 0.4 \textbf{ AND } i - z > 0.4$

Expected Factorization: {u, g}, {g, r}, {r, i}, {i, z}

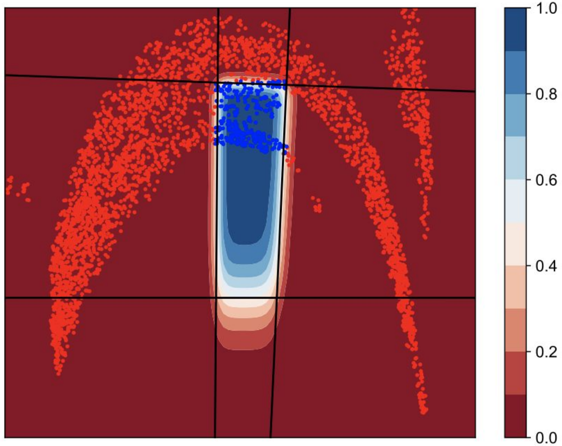
Computed Factorization: {u, g}, {g, r}, {r, i}, {i, z}

Identical

Penalized FLM in Practice - SDSS Query 05



{Rowc, Colc} subspace
Circular pattern



{Ra, Dec} subspace
Rectangular pattern

User interest is modeled as a **low-dimensional convex object** in each subspace

Attributes: *rowc, colc, ra, dec*

Query predicate: $(rowc - 682.5)^2 + (colc - 1022.5)^2 < 90^2$ **AND** $180 < ra < 210$ **AND** $50 < dec < 60$

Expected Factorization: $\{rowc, colc\}, \{ra\}, \{dec\}$

Computed Factorization: $\{rowc, colc\}, \{ra\}, \{dec\}$

Identical

Experimental Evaluation - Batch Case

Batch setting: SDSS queries, 1.8 million tuples, 50% - 50% train test split

F-score

Query	SVM	FLM	VIPR [NIPS2012]
05	80	79	73
06	18	0	10
07	90	47	72
08	98	94	88
09	92	100	75
10	89	86	41
11	83	84	71

Time (min)

Query	SVM	FLM	VIPR [NIPS2012]
Q5	0.4	3.8	0.5
Q6	16.1	3.8	2.2
Q7	1158.2	3.8	0.6
Q8	45.7	4.2	3.3
Q9	102.9	3.7	1.2
Q10	10.4	3.8	1.2
Q11	3.5	4.1	1.2

➔ FLM approximates the performance of SVM, while being **interpretable**

➔ FLM outperforms VIPR, another interpretable model

➔ FLM can be more efficient for training than SVM

Extension to the Active Learning Scenario

Uncertainty Sampling: select the point which the current model is "most uncertain":

$$\arg\min_{x \in \mathcal{U}} |p_{b,W}(x) - 0.5|$$

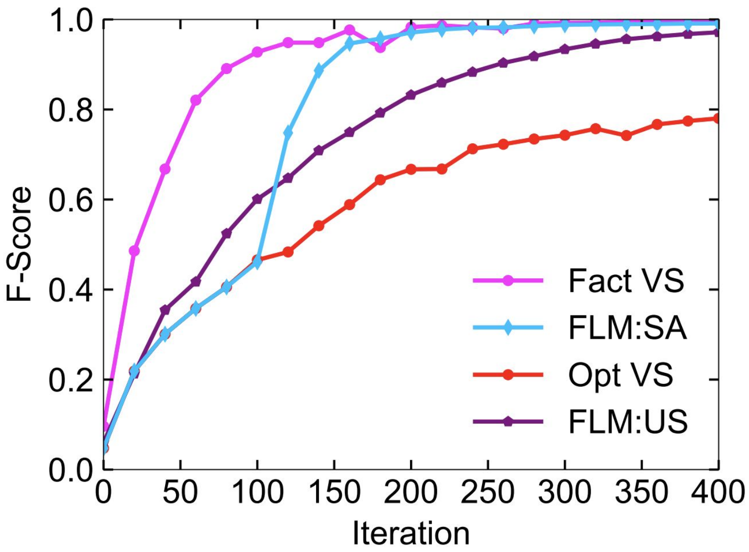
Problem
Uncertainty Sampling can be myopic in its selection

Swapping Algorithm: start with OptVS, then swap to FLM-based Uncertainty Sampling

➔ Enjoys the fast initial convergence of VS-based methods

➔ In later iterations, can profit of FLM's enhanced classification accuracy

Experimental Evaluation - AL Case



F-score for SDSS 09 (1.5%, 5D)

Algorithms

- **FLM:SA**: Swapping Algorithm
- **FLM:US**: FLM + Uncertainty Sampling
- **FactVS**: factorized AL. Leverages extra information from the user (factorization, subspace labels)
- **OptVS**: baseline non-factorized AL

The Swapping Algorithm approximates FactVS while outperforming non-factorized learners

Presentation Outline

1. Introduction
2. Version Space Algorithms
3. A Factorized Version Space Algorithm
4. Learning a Factorized Classifier
5. **Related Work**
6. Summary

Version Space Algorithms

- **Active Learning under Margin Assumptions (ALuMA)**
  - Sampling-based approximation of GBS for linear classifiers
  - Extension for kernel classifiers, but is often too costly in practice
  - Strong theoretical guarantees on performance (similar as GBS)
- **Simple Margin (SM)**
  - SVM-based Uncertainty Sampling strategy, shown to approximately bisect the VS
- **Query-by-Disagreement (QBD)**
  - Approximates the VS by a positive and a negatively biased classifiers

Kernel Query-by-Committee (KQBC): R. Gilad-Bachrach, A. Navot, and N. Tishby. Query-by-committee made real. In *Advances in Neural Information Processing Systems 18,* 2006.
ALuMA: A. Gonen, S. Sabato, and S. Shalev-Shwartz. Efficient active learning of halfspaces: an aggressive approach. *Journal of Machine Learning Research*, 14(1): 2583–2615, 2013
Simple Margin (SM): S. Tong and D. Koller. Support Vector Machine active learning with applications to text classification. *Journal of Machine Learning Research*, 2:45–66, 2001.
Query-by-Disagreement (QBD): B. Settles. *Active Learning*. Morgan & Claypool Publishers, 2012.

Human-in-the-Loop Model Development

Data Programming under Weak Supervision

- **Snorkel**
  - **Labeling Function (LF)**: simple heuristics used for labeling data instances (yes, no, unknown)
  - An accurate classifier can be built without users labeling a single data point
- **Snuba**
  - Automatic generation of LFs by relying on a “small” labeled set
  - In practice, may still require thousands of labeled examples to reach high accuracy

Interactive Data Exploration

- **Dual Space Model (DSM)**
  - Convexity: user interest region (or its complement) is convex
  - Polytope model: Automatically labels data examples through a data-space decomposition
  - Can also leverages a factorization from the user

Snorkel: A. J. Ratner, C. M. De Sa, S. Wu, D. Selsam, and C. Ré. Data programming: Creating large training sets, quickly. In Advances in Neural Information Processing Systems 29, 2016.
Snuba: P. Varma and C. Ré. Snuba: Automating weak supervision to label training data. Proceedings of the VLDB Endowment, 12(3):223–236, 2018.
Dual-Space Model (DSM): E. Huang, L. Peng, L. D. Palma, A. Abdelkafi, A. Liu, and Y. Diao. Optimization for active learning-based interactive database exploration. *Proceedings of the VLDB Endowment*, 12(1):71–84, 2018.

Interpretable ML and Feature Selection

Interpretable ML

- VIPR
  - Assumes that any data point can be locally classified by a small number of features
  - Computes a mapping of data points and low-dimensional projections
  - **Contrast to FLM**: we assume the final decision is a combination of low dimensional predictions

Lasso-based Feature Selection

- Lasso
  - Improves the interpretability of linear classifiers in high-dimensions
  - Forces the weight vector to be sparse
- **Group Lasso**
  - Allows for the selection of groups of features at a time
  - Useful for selecting categorical variables

VIPR: M. Fiterau and A. Dubrawski. Projection retrieval for classification. In Advances in Neural Information Processing Systems, volume 25. Curran Associates, Inc., 2012
Lasso: R. Tibshirani. Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society (Series B), 58:267–288, 1996.
Group Lasso: M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. Journal of the Royal Statistical Society: Series B, 2006
Group Lasso for Categorical Variables: J. Chiquet, Y. Grandvalet, and G. Rigaill. On coding effects in regularized categorical regression. Statistical Modelling, 2016

Presentation Outline

1. Introduction
2. Version Space Algorithms
3. A Factorized Version Space Algorithm
4. Learning a Factorized Classifier
5. Related Work
6. **Summary**

Summary

To overcome the slow start problem in AL, we developed the following contributions:

- **OptVS**: An optimized VS algorithm providing:
  - Strong *theoretical guarantees* on performance
  - An *efficient implementation* in time and space
- **FactVS**: A factorized VS algorithm which leverages extra information from the user to further expedite convergence
- **FLM**: A learning algorithm for factorized classifiers, capable of decomposing the user interest as a combination of low-dimensional convex objects
- **Swapping Algorithm**: an automatically factorized AL strategy that leverages both OptVS and FLM to provide an effective data exploration strategy

Thank you! Questions?

List of Publications

[1] E. Huang, L. Di Palma, L. Cetinsoy, Y. Diao, and A. Liu. AIDEme: An active learning based system for interactive exploration of large datasets, *Neural Information Processing Systems (NeurIPS)*, Demonstration Track, Dec 2019

[2] L. Di Palma, Y. Diao, and A. Liu. A Factorized Version Space Algorithm for Human-in-the-Loop Data Exploration, *International Conference in Data Mining (ICDM)*, Nov 2019

[3] E. Huang, L. Peng, L. Di Palma, A. Abdelkafi, A. Liu, and Y. Diao. Optimization for Active Learning-based Interactive Database Exploration, *Proceedings of the VLDB Endowment (PVLDB)*, 2018