IJCAI-2015 tutorial, July 25, 2015, Buenos Aires, Argentina

Lifelong Machine Learning
in the Big Data Era

Zhiyuan Chen and Bing Liu

Department of Computer Science
University of Illinois at Chicago

czyuanacm@gmail.com, liub@cs.uic.edu

Introduction

- Classic Machine Learning (ML) paradigm (isolated single-task learning): Given a dataset, run an ML algo. to build a model
  - without considering any related information or the past learned knowledge – learning in isolation
- Existing ML algorithms such as
  - SVM, NB, DT, Deep NN, CRF, and topic models
  - have been very successful in practice.
- Let's call this: Machine Learning (ML) 1.0.

Introduction: ML 1.0

(Thrun, 1996b; Silver et al 2013; Chen and Liu, 2014a)

- But such "isolated learning" has weaknesses.
  - No memory: Knowledge learned is not retained.
    - Knowledge is not cumulative.
    - Cannot learn by leveraging past learned knowledge
  - Needs a large number of training examples.
    - Humans can learn effectively from a few examples.
- Humans never learn in isolation.
- Probably not possible to build an intelligent agent using only ML 1.0 algorithms.

Introduction: ML 2.0

- Learn as humans do.
  - *lifelong machine learning* (LML)
  - Retain learned knowledge from previous tasks & use it to help future learning
- Let us call this paradigm Machine Learning 2.0
  - LML is likely to need a systems approach
- Big data provides a great opportunity for LML
  - E.g., big text data from social media
  - Extensive sharing of concepts across tasks/domains due to the nature of the natural language

A Large Space

- Many relevant topics and problems
  - Transfer learning or domain adaptation
  - Multitask learning (batch and online)
  - Lifelong learning
  - Never-ending learning
  - Continual learning
  - Cumulative learning
  - …
- It reflects the richness & diversity of learning
  - LML is sometimes considered too wide a field and confusing (Silver et al., 2013)

Plan for the Tutorial

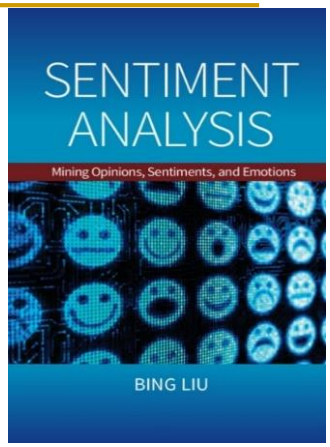
- Since there are many relevant topics and
  - some of them are very large themselves, e.g., *transfer learning* and *multitask learning*,
    - There are focused tutorials about them
  - impossible to cover all problems/techniques
- After the definition of LML,
  - Selectively cover some representative or example papers in several main topics.
  - Focus: topics and papers that match well with the LML definition

Outline

- Introduction
- A motivating example
- What is lifelong learning?
- Transfer learning
- Multitask learning
- Supervised lifelong learning
- Semi-supervised never-ending learning
- Unsupervised lifelong topic modeling
- Summary

A Motivating Example

(Liu, 2012; 2015)



- Sentiment analysis or opinion mining
  - computational study of opinion, sentiment, appraisal, evaluation, attitude, and emotion.
- Active research area in NLP and unlimited applications
  - Useful to every organization and individual.
- Suitable for LML
  - extensive knowledge sharing across tasks/domains
    - Sentiment expressions, e.g., good, bad, expensive, great.
    - Sentiment targets, e.g., "*The screen is great*."

(1) Sentiment Classification

- "*I bought an iPhone a few days ago. It is such a nice phone. The touch screen is really cool. The voice quality is great too. ....*"
- Goal: classify docs or sentences as + or -.
  - Need to manually label a lot of training data for each domain, which is highly labor-intensive
  - Can we not label for every domain or at least not so many docs/sentences?

Exploiting the Past Information

- It is "well-known" that a sentiment classifier (SC) built for domain A will not work for domain B.
  - E.g., SC built for "camera" will not work for "earphone"
- Classic solution: transfer learning
  - Using labeled data in the past domain S (camera) to help learning in the target domain T (earphone).
  - If S and T are very similar, S can help.
- This may not be the best solution!

Lifelong Sentiment Classification

(Chen, Ma and Liu 2015)

Imagining - we have worked on a *large number of past domains/tasks* with their training data D.

- do we need any data from the new domain T?

- No in many cases – A naive "*LML*" method by polling all data together works wonders.
  - Can improve accuracy by as much as 19% (= 80%-61%)
  - Why? Sharing of sentiment expressions
- Yes in other cases: e.g., we build a SC using D, but it works poorly for toy reviews.
  - Why? Because of the word "toy"

(2) Lifelong Aspect Extraction

(Chen and Liu, 2014a, 2014b)

- "*The battery life is long, but pictures are poor*."
  - Aspects (opinion targets): battery life, picture
- Observation:
  - A fair amount of aspect overlapping across reviews of different products or domains
    - Every product review domain has the aspect *price*,
    - Most electronic products share the aspect *battery*
    - Many also share the aspect of *screen*.
  - It is rather "silly" not to exploit such sharing in learning or extraction.

Lifelong Topic Modeling

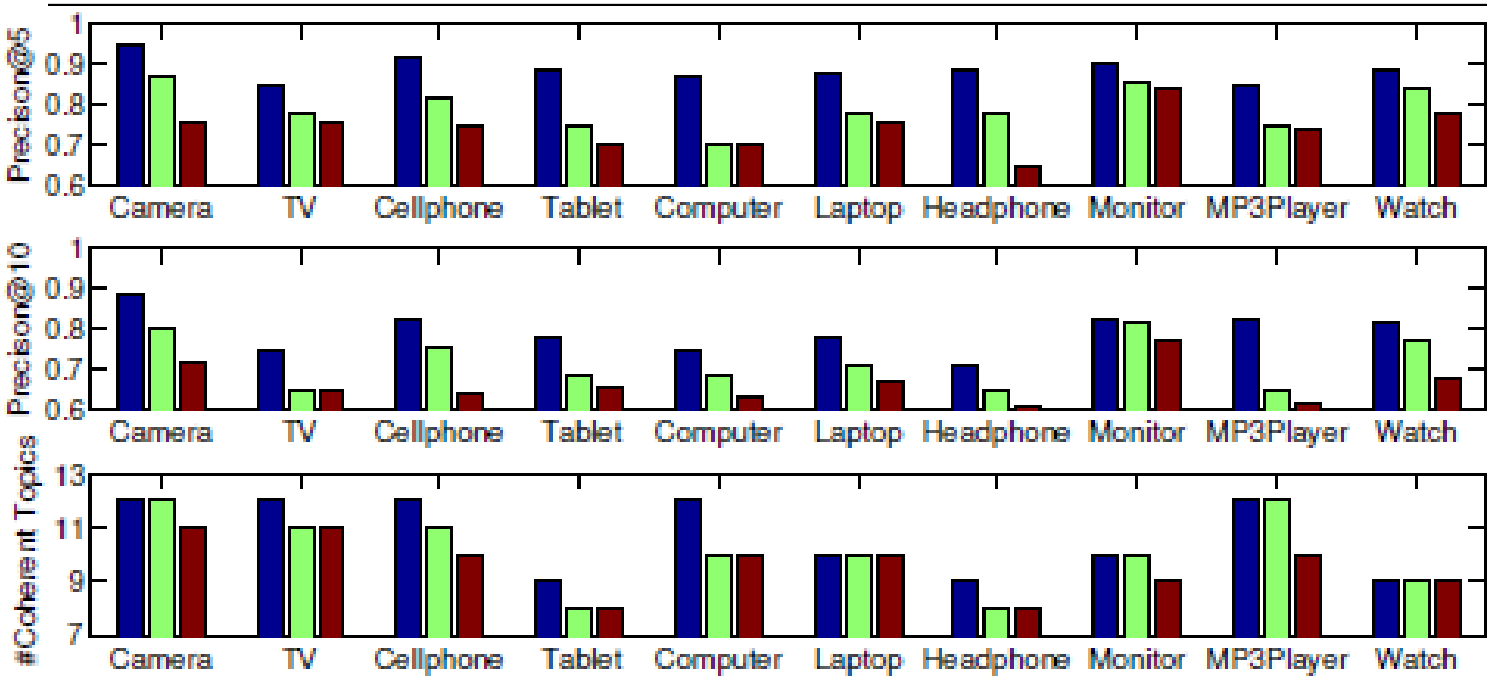


Figure 2. Top & Middle: Topical words *Precision*@5 & *Presicion*@10 of coherent topics of each model respectively; Bottom: number of coherent (#Coherent) topics discovered by each model. The bars from left to right in each group are for LTM, LDA, and DF-LDA. On average, for *Precision*@5 and

Outline

- Introduction
- A motivating example
- What is lifelong learning?
- Transfer learning
- Multitask learning
- Supervised lifelong learning
- Semi-supervised never-ending learning
- Unsupervised lifelong topic modeling
- Summary

Lifelong Learning (LML) definition

- Definition: The learner has performed learning on a sequence of tasks, from 1 to *N*-1. When faced with the *N*th task, it uses the relevant knowledge gained in the past *N* -1 tasks to help learning for the *N*th task.
  - An LML system thus needs four components:
    - Past Information Store (PIS)
    - Knowledge Base (KB)
    - Knowledge Miner (KM)
    - Knowledge-Based Learner (KBL)

Past Information Store (PIS)

- It stores the information from the past learning. It may have sub-stores for storing information such as
  - the original data used in each past task,
  - the intermediate results from the learning of each past task,
  - the final model or patterns learned from each past task
  - etc.

Knowledge Base (KB)

- It stores the knowledge mined/consolidated from PIS (Past Information Store).
  - Meta-knowledge discovered from PIS, e.g., general/shared knowledge applicable to multiple domains/tasks.
    - E.g., a list of words commonly used to represent positive or negative sentiment.
  - This requires a general knowledge representation scheme suitable for a class of applications.

Knowledge Miner (KM)

- It mines (meta) knowledge from PIS (Past Information Store).
- This mining is regarded as a meta-mining process because it learns knowledge from information resulted from learning of the past tasks.
- The resulting knowledge is stored to KB (Knowledge Base).

Knowledge-Based Learner (KBL)

- Given the knowledge in KB, the LML learner can leverage the knowledge and possibly some information in PIS to learn from the new task, which should
  - Learn better even with a large amount of training data
  - Learn well with a small amount of data
  - …

LML: Flexible Learning

- It can use any past knowledge or information in any way to help the new task learning.
- It can focus on learning the Nth task by using knowledge gained from the past N-1 tasks.
- It can also improve any of the models from the past N–1 tasks based on results from the other N-1 tasks (including the Nth task):
  - By treating that previous task as the “Nth” task.

Outline

- Introduction
- A motivating example
- What is lifelong learning?
- Transfer learning
- Multitask learning
- Supervised lifelong learning
- Semi-supervised never-ending learning
- Unsupervised lifelong topic modeling
- Summary

Transfer Learning

- Transfer learning has been studied extensively - survey by Pan & Yang (2010).
- Problem statement:
  - Source domain(s) (usually 1 source domain/task)
    - With labeled training data
  - Target domain (assume to be *related*)
    - With little or no labeled training data but unlabeled data
  - Goal: leverage the information from the source domain(s) to help learning in the target domain
    - Only optimize the target domain/task learning

Transfer Learning as LML

- Transfer learning can be regarded as a special case of LML

- **PIS**: mainly store the data from the source domain(s).
- **KM**: It generates the knowledge from the source domain data dynamically based on the target domain unlabeled data to be transferred to the target domain.

KB & KBL of Transfer Learning

- (Bickel et al., 2007; Sugiyama et al., 2008; Liao et al., 2005; Dai et al., 2007b, 2007c; Jiang & Zhai 2007)
  - **KB**: Some data instances in the source domain
  - **KBL**: Instance reweighting or Important sampling
- (Ando & Zhang, 2005; Dai et al., 2007a; Daume III, 2007; Blitzer et al., 2006; 2007; Wang & Mahadevan, 2008)
  - **KB**: Features from source domain
  - **KBL**: use KB to generate new features for target dom.

One Transfer Learning Technique

- Structural correspondence learning (SCL) (Blitzer et al 2006)
- Identify correspondences among features from different domains by modeling their correlations with some pivot features.
  - Pivot features are features which behave in the same way for learning in both domains.
  - Non-pivot features from different domains which are correlated with many of the same pivot features are assumed to correspond.

SCL (contd)

- SCL works with a source domain and a target domain. Both domains have ample unlabeled data, but only the source has labeled data.
- SCL first chooses a set of *m* features which occur frequently in both domains (and are also good predictors of the source label).
- These features are called the *pivot features* which represent the shared feature space of the two domains.

Choose Pivot Features

- For different applications, pivot features may be chosen differently, for example,
  - For part-of-speech tagging, frequently-occurring words in both domains were good choices (Blitzer et al., 2006)
  - For sentiment classification, features are words that frequently-occur in both domains and also have high mutual information with the source label (Blitzer et al., 2007).

Finding Feature Correspondence

- Compute the correlations of each pivot feature with non-pivot features in both domains by building binary pivot predictors

$$f_\ell(\mathbf{x}) = \mathrm{sgn}(\hat{\mathbf{w}}_\ell \cdot \mathbf{x}), \quad \ell = 1 \ldots m$$

  - using unlabeled data (predicting whether the pivot feature *l* occurs in the instance.)
  - The weight vector $\hat{\mathbf{w}}_\ell$ encodes the covariance of the non-pivot features with the pivot feature

Finding Feature Correspondence

- Positive values in $\hat{\mathbf{w}}_\ell$:
  - indicate that those non-pivot features are positively correlated with the *l*-th pivot feature in the source or the target,
  - establish a feature correspondence between the two domains.
- Produce a correlation matrix *W*

$$W = [\hat{\mathbf{w}}_1 | \ldots | \hat{\mathbf{w}}_m],$$

Compute Low Dim. Approximation

- Instead of using *W* to directly create *m* extra features.
- SVD(*W*) = *U D* V^T is employed to compute a low-dimensional linear approximation $\theta = \tilde{U}^T_{[1:h,:]}$ (the top *h* left singular vectors).
- The final set of features used for training and for testing is the original set of features **x** combined with θ**x**.

SCL Algorithm

Input: labeled source data $\{(\mathbf{x}_t, y_t)_{t=1}^T\}$,
unlabeled data from both domains $\{\mathbf{x}_j\}$

Output: predictor $f : X \rightarrow Y$

1. Choose m pivot features. Create m binary prediction problems, $p_\ell(\mathbf{x})$, $\ell = 1 \ldots m$

2. For $\ell = 1$ to m

$$\hat{\mathbf{w}}_\ell = \underset{\mathbf{w}}{\operatorname{argmin}} \left(\sum_j L(\mathbf{w} \cdot \mathbf{x}_j, p_\ell(\mathbf{x}_j)) + \lambda ||\mathbf{w}||^2 \right)$$

end

3. $W = [\hat{\mathbf{w}}_1 | \ldots | \hat{\mathbf{w}}_m]$, $[U\ D\ V^T] = \text{SVD}(W)$,
$\theta = U^T_{[1:h,:]}$

4. Return f, a predictor trained on $\left\{ \left(\begin{bmatrix} \mathbf{x}_t \\ \theta \mathbf{x}_i \end{bmatrix}, y_t \right)_{t=1}^T \right\}$

A Simple EM Style Approach

(Rigutini, 2005; Chen et al, 2013)

- The approach is similar to SCL
  - Pivot features are selected through feature selection on the labeled source data
- Transfer is done iteratively in an EM style using naïve Bayes
  - Build an initial classifier based on the selected features and the labeled source data
  - Apply it on the target domain data and iteratively perform knowledge transfer with the help of feature selection.

The Algorithm

Algorithm EM-Transfer

Input: Labeled source data D_L and unlabeled target data D_U

1 Select a set Δ of k_1 features from D_L using IG;
2 Learn an initial naïve Bayes classifier h from D_L based on Δ
3 **repeat**
4 **for** each document d_i in D_U **do**
5 $c = h(d_i)$; // predict the class of d_i using h
6 Produce data D_P based on predicted class of D_U;
7 Select a new set Δ of k_2 features from D_P;
8 Learn a new classifier h on D_P based on the new Δ;
9 **until** the predicted classes of D_U stabilize
10 Return the classifier h from the last iteration.

A Large Body of Literature

- Transfer learning has been a popular research topic and researched in many fields, e.g.,
  - Machine learning
  - data mining
  - NLP
  - vision
- Pan & Yang (2010) presented an excellent survey with extensive references.

Outline

- Introduction
- A motivating example
- What is lifelong learning?
- Transfer learning
- Multitask learning
- Supervised lifelong learning
- Semi-supervised never-ending learning
- Unsupervised lifelong topic modeling
- Summary

Multitask Learning (MTL)

- Problem statement: Co-learn multiple related tasks simultaneously:
  - All tasks have labeled data and are treated equally
  - Goal: optimize learning/performance across all tasks through shared knowledge
- Rationale: introduce inductive bias in the joint hypothesis space of all tasks (Caruana, 1997)
  - by exploiting the task relatedness structure, or
  - shared knowledge

Compared with Other Problems

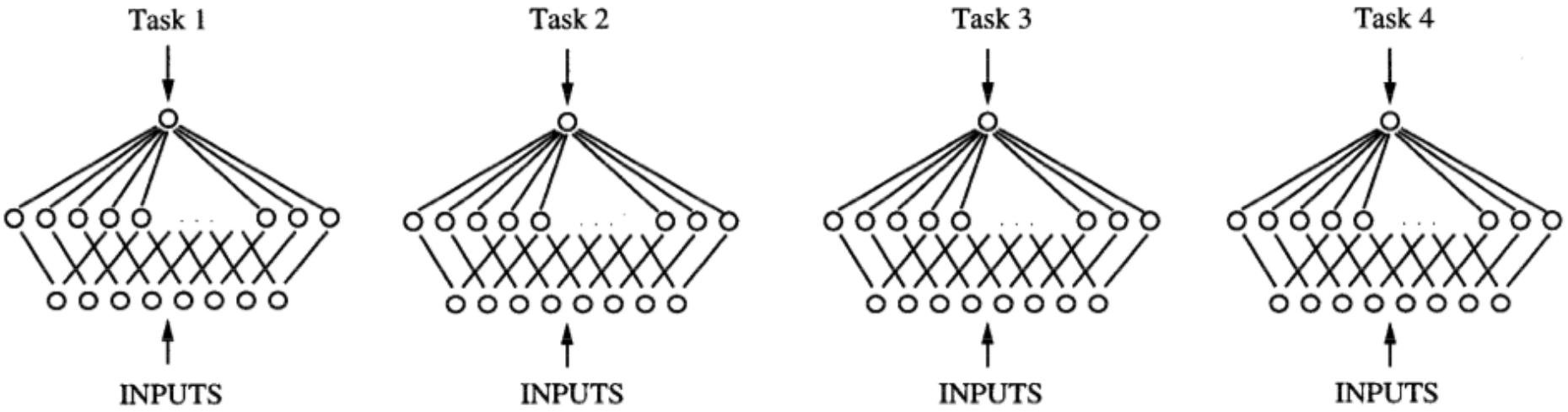
Giving a set of learning tasks, t_1, t_2, …, t_n

- Single task learning: learn each independently
  - $\min_{w_1} L_1$, $\min_{w_2} L_2$, …, $\min_{w_n} L_n$
- Multitask learning: co-learn all simultaneously
  - $\min_{w_1, w_2, \ldots, w_n \in \Omega} \frac{1}{n} \sum_{i=1}^{n} L_i$
- Transfer learning: Learn well only on the target task. Do not care about learning of the source.
  - Target domain/task has little or no labeled data
- *Lifelong learning*: help learn well on future target tasks, without seeing future task data (??)

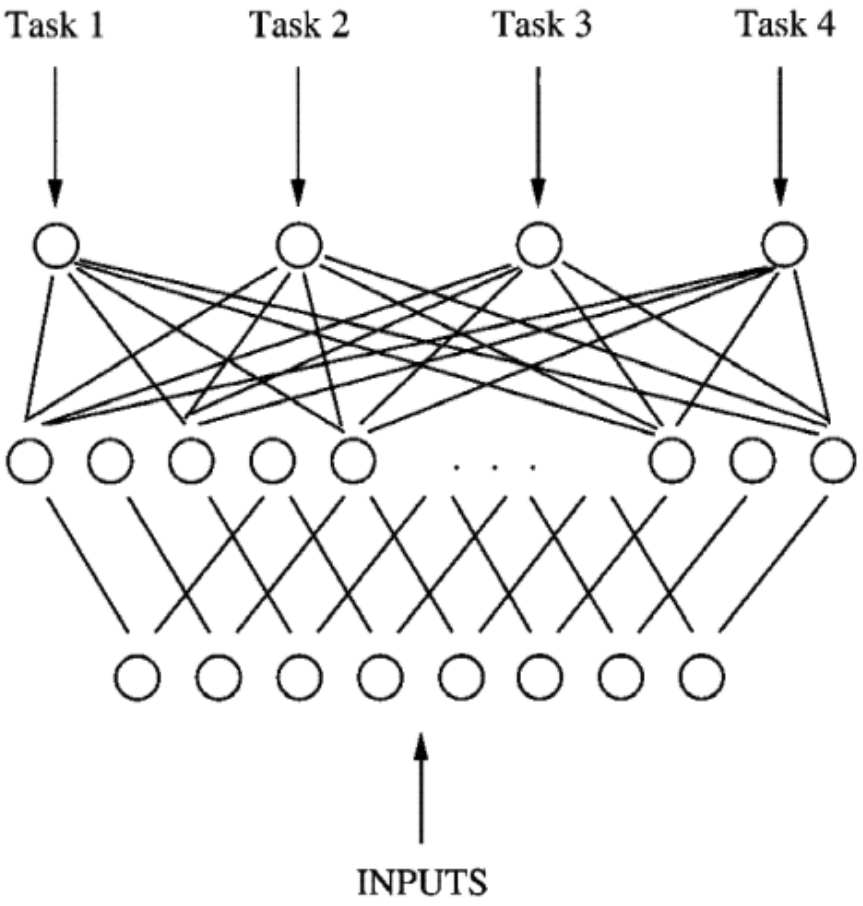
Multitask Learning in (Caruana, 1997)

- Since model trained for a single task may not generalize well, due to lack of training data,
  - The paper performs multitask learning using artificial neural network
- Multiple tasks share a common hidden layer
  - One combined input for the neural nets
  - One output unit for each task
  - Back-propagation is done in parallel on the all outputs in the MTL net.

Single Task Neural Nets



MTL Neural Network



Results of MTL Using Neural Nets

- Pneumonia Prediction

Table 3. Error Rates (fraction deaths) for STL with Rankprop and MTL with Rankprop on Fractions of the Population (FOP) predicted to be at low risk between 0.0 and 0.5. MTL makes 5–10% fewer errors than STL.

FOP	0.1	0.2	0.3	0.4	0.5
STL Rankprop	.0083	.0144	.0210	.0289	.0386
MTL Rankprop	.0074	.0127	.0197	.0269	.0364
% Change	-10.8%	-11.8%	-6.2% *	-6.9% *	-5.7% *

MTL for kNN

- The paper also proposed MTL for kNN with

$$Distance(case) = \sqrt{\sum_{i=1}^{NO\_ATTS} weight_i * (\Delta attribute_i)^2}$$

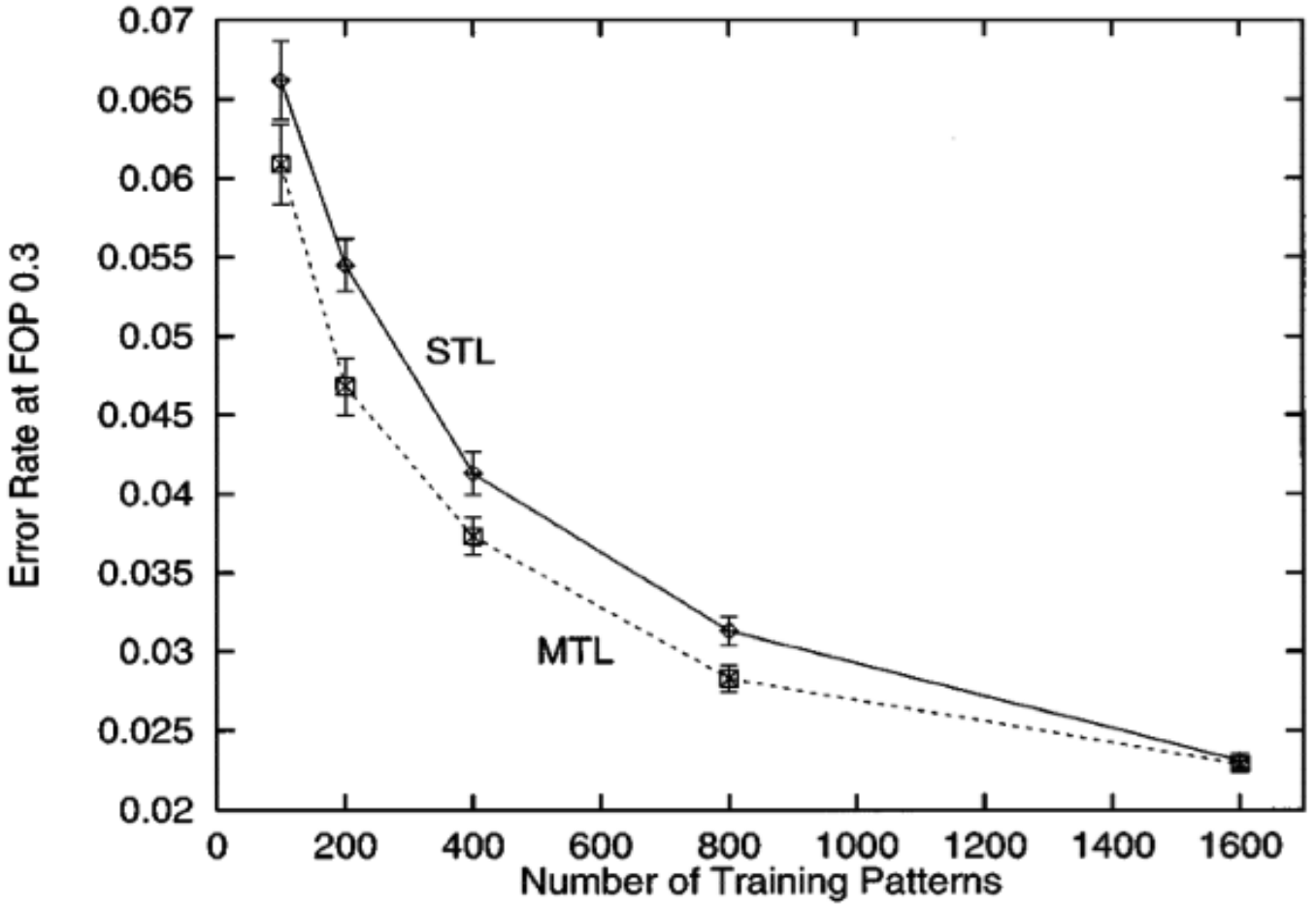
- Its uses the performances on multiple tasks for optimization to choose the weights.

$$Eval\_Metric = Perf\_Main\_Task + \sum_{i=1}^{NO\_TASKS} \lambda_i * Perf\_Extra\_Task_i$$

  - $\lambda_i = 0$: ignore the extra/past tasks,
  - $\lambda_i \approx 1$: treat all tasks equally.
  - $\lambda_i \gg 1$: more attention to extra tasks than main task.
- *More like to lifelong learning*

One Result of MTL for kNN

- Pneumonia Prediction



GO-MTL Model

(Kumar et al., ICML-2012)

- Most multitask learning methods assume that all tasks are related. But this is not always the case in applications.
- GO-MTL: Grouping and Overlap in Multi-Task Learning
- The paper first proposed a general approach and then applied it to
  - regression and classification
  - using their respective loss functions.

Notations

- Given *T* tasks in total, let

$Z_t = \{(\mathbf{x}_{ti}, y_{ti}) : i = 1, 2, \ldots, N_t\}$ be the training set for each task $t = 1, 2, \ldots, T$. Let $\mathbf{w}_t$ represent the weight vector for task indexed by t. These task weight vectors are stacked as columns of a matrix $\mathbf{W}$, which is of size $d \times T$, with d being the feature dimension.

- The initial ***W*** is learned from *T* individual tasks.
  - E.g., weights/parameters of linear regression or logistic regression

The Approach

We assume there are $k(< T)$ latent basis tasks and each observed task can be represented as linear combination of a *subset* of these basis tasks. This assumption enables us to write the weight matrix $\mathbf{W}$ as $\mathbf{W} = \mathbf{LS}$, where $\mathbf{L}$ is a matrix of size $d \times k$ with each column representing a latent task, and $\mathbf{S}$ is a matrix of size $k \times T$ containing the weights of linear combination for each task. The predictor $\mathbf{w}_t$ for task t is given by $\mathbf{Ls}_t$, where $\mathbf{s}_t$ is t'th column of matrix $\mathbf{S}$.

- **S** is assumed to be sparse. **S** also captures the task grouping structure.

Optimization Objective Function

If $\mathbf{s}_t$ denotes the sparsity pattern for task t, our learning cost function takes the following form:

$$\sum_t \sum_{(\mathbf{x}_{ti}, y_{ti}) \in Z_t} \mathcal{L}\left(y_{ti}, \mathbf{s}_t' \mathbf{L}' \mathbf{x}_{ti}\right) + \mu ||\mathbf{S}||_1 + \lambda ||\mathbf{L}||_F^2, \quad (1)$$

where $\mathcal{L}(\cdot, \cdot)$ is the empirical loss function, $|| \cdot ||_1$ is entry-wise ℓ_1 norm of the matrix and $||\mathbf{L}||_F = (tr(\mathbf{L}\mathbf{L}'))^{1/2}$ is the Frobenius norm of matrix $\mathbf{L}$. The parameter μ controls the sparsity in $\mathbf{S}$. The penalty on the Frobenius norm of $\mathbf{L}$ regularizes the predictor weights to have low ℓ_2 norm and avoids overfitting.

Optimization Strategy

- Alternating optimization strategy to reach a local minimum.
- For a fixed **L**, optimize s_t :

$$\mathbf{s}_t = \arg\min_{\mathbf{s}} \sum_{(\mathbf{x}_{ti}, y_{ti}) \in Z_t} \mathcal{L}\left(y_{ti}, \mathbf{s}'\mathbf{L}'\mathbf{x}_{ti}\right) + \mu||\mathbf{s}||_1, \quad (2)$$

- For a fixed **S**, optimize **L**:

$$\min_{\mathbf{L}} \sum_{t=1}^{T} \sum_{(\mathbf{x}_{ti}, y_{ti}) \in Z_t} \mathcal{L}\left(y_{ti}, \mathbf{s}_t'\mathbf{L}'\mathbf{x}_{ti}\right) + \lambda||\mathbf{L}||_F^2. \quad (3)$$

GO-MTL Algorithm

Input:
Z^t: Labeled training data for all tasks
k: Number of latent tasks
μ: Parameter for controlling sparsity
Output: Task predictor matrix $\mathbf{W}$, $\mathbf{L}$ and $\mathbf{S}$.
1: Learn individual predictors for each task using only its own data.
2: Let $\mathbf{W}^0$ be the matrix that contains these initial predictors as columns.
3: Compute top-k singular vectors: $\mathbf{W}^0 = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$
4: Initialize $\mathbf{L}$ to first k columns of $\mathbf{U}$.
while *not converged* **do**
  for $t = 1$ to T **do**
    5: Solve Eq. 2 to obtain $\mathbf{s}_t$.
  end for
  6: Construct matrix $\mathbf{S} = [\mathbf{s}_1 \mathbf{s}_2 \ldots \mathbf{s}_T]$.
  7: Save the previous $\mathbf{L}$: $\mathbf{L}_{\text{old}} = \mathbf{L}$.
  8: Fix $\mathbf{S}$ and solve Eq. 3 to obtain $\mathbf{L}$.
end while
9: Return outputs: $\mathbf{L} = \mathbf{L}_{\text{old}}$, $\mathbf{S}$ and $\mathbf{W} = \mathbf{L}_{\text{old}}\mathbf{S}$.

One Result

	Synth. (1)	Synth. (2)	Computer	School	MNIST	USPS
STL	1.04	1.36	2.70 (0.10)	10.67 (0.20)	14.8 (0.34)	9.0 (0.4)
No-group MTL	0.48	0.79	2.06 (0.07)	10.18 (0.15)	14.4 (0.28)	7.8 (0.2)
DG-MTL	0.42	0.80	2.01 (0.10)	10.18 (0.20)	14.0 (0.30)	7.8 (0.2)
GO-MTL	**0.35**	**0.64**	**1.76** (0.09)	10.04 (0.24)	**13.4** (0.30)	**7.2** (0.2)

Table 1. Results on different datasets: Reported numbers are root mean square error (RMSE) for regression datasets and multi-class classification errors for MNIST and USPS. Numbers in parentheses are std. dev., which were negligible for synthetic datasets and so are not reported. STL: Single task learning, No-group MTL (Argyriou et al., 2008a), DG-MTL (Kang et al., 2011), GO-MTL: the proposed method.

A Large Body of Literature

- Two tutorials on MTL
  - Multi-Task Learning: Theory, Algorithms, and Applications. SDM-2012, by Jiayu Zhou, Jianhui Chen, Jieping Ye
  - Multi-Task Learning Primer. IJCNN'15, by Cong Li and Georgios C. Anagnostopoulos

Various task assumptions and models:

- All tasks share a common parameter vector with a small perturbation for each (Evgeniou & Pontil, 2004)
- Tasks share a common underlying representation (Baxter 2000; Ben-David & Schuller, 2003)
- Parameters share a common prior (Yu et al., 2005; Lee et al., 2007; Daume´ III, 2009).

MTL Assumptions and Models

- A low dimensional representation shared across tasks (Argyriou et al., 2008).
- Tasks can be clustered into disjoint groups (Jacob et al., 2009; Xue et al., 2007).
- The related tasks are in a big group while the unrelated tasks are outliers (Yu et al., 2007; Chen et al., 2011)
- The tasks were related by a global loss function (Dekel et al., 2006)
- Task parameters are a linear combination of a finite number of underlying bases (Kumar et al., 2012; Ruvolo & Eaton, 2013a)
- Lawrence and Platt (2004) learn the parameters of a shared covariance function for the Gaussian process

Some Online MTL techniques

- Multi-Task Infinite Latent Support Vector Machines (Zhu, J. et al., 2011)
- Joint feature selection (Zhou et.al. 2011)
- Online MTL with expert advice (Abernethy et al., 2007, Agarwal et al., 2008)
- Online MTL with hard constraints (Lugosi et al., 2009)
- Reducing mistake bounds for the online MTL (Cavallanti et al., 2010)
- Learn task relatedness adaptively from the data (Saha et al., 2011)
- Method for multiple kernel learning (Li et al. 2014)

MTL with Applications

- Web Pages Categorization (Chen et al., 2009)
- HIV Therapy Screening (Bickel et al., 2008)
- Predicting disease progression (Zhou et al., 2011)
- Compiler performance prediction problem based on Gaussian process (Bonilla et al., 2007)
- Visual Classification and Recognition (Yuan et al., 2012)

Outline

- Introduction
- A motivating example
- What is lifelong learning?
- Transfer learning
- Multitask learning
- Supervised lifelong learning
- Semi-supervised never-ending learning
- Unsupervised lifelong topic modeling
- Summary

Early Work on Lifelong Learning
(Thrun, 1996b)

- Concept learning tasks: The functions are learned over the lifetime of the learner, f_1, f_2, f_2, … $\in F$.
- Each task: learn the function $f: I \rightarrow \{0, 1\}$. f(x)=1 means x is a particular concept.
  - For example, f_{dog}(x)=1 means x is a dog.
- For nth task, we have its training data X
  - Also the training data X_k of k =1 , 2, …, n-1 tasks. X_k is called a *support set* for X.

Intuition

- The paper proposed a few approaches based on two learning algorithms,
  - Memory-based, e.g., kNN or Shepard method
  - Neural networks,
- Intuition: when we learn $f_{dog}(x)$, we can use functions or knowledge learned from previous tasks, such as $f_{cat}(x)$, $f_{bird}(x)$, $f_{tree}(x)$, etc.
  - Data for $f_{cat}(X)$, $f_{bird}(X)$, $f_{tree}(X)$… are *support sets*.

Memory based Lifelong Learning

- First method: use the support sets to learn a new representation, or function

  g: $I \rightarrow I'$

  - which maps input vectors to a new space. The new space is the input space for the final *k*NN.
  - Adjust *g* to minimize the energy function.

$$E := \sum_{k=1}^{n-1} \sum_{\langle x, y=1 \rangle \in X_k} \left(\sum_{\langle x', y'=1 \rangle \in X_k} ||g(x) - g(x')|| - \sum_{\langle x', y'=0 \rangle \in X_k} ||g(x) - g(x')|| \right)$$

  - g is a neural network, trained with Back-Prop. kNN or Shepard is then applied for the nth task

Second Method

- It learns a distance function using support sets

  d: $I \times I \rightarrow [0, 1]$

  - It takes two input vectors x and x' from a pair of examples <x, y>, <x', y'> of the same support set X_k (k = 1, 2, , …, n-1)
  - d is trained with neural network using back-prop, and used as a general distance function
  - Training examples are:

$$\langle (x, x'), 1 \rangle \quad \text{if } y{=}y'{=}1$$

$$\langle (x, x'), 0 \rangle \quad \text{if } (y{=}1 \wedge y'{=}0) \text{ or } (y{=}0 \wedge y'{=}1)$$

Making Decision

- Given the new task training set X_n and a test vector x, for each +ve example, (x', y'=1)$\in X_n$,
  - d(x, x') is the probability that x is a member of the target concept.
- Decision is made by using votes from positive examples, <x_1, 1>, <x_2, 1>, … $\in X_n$ combined with Bayes' rule

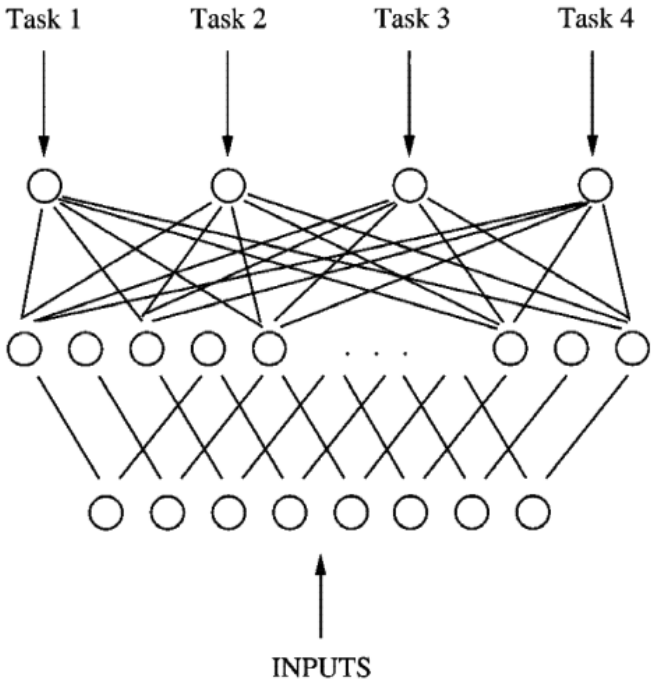
$$P(f_n(x)=1) = 1 - \left(1 + \prod_{\langle x', y'=1\rangle \in X_n} \frac{d(x,x')}{1-d(x,x')}\right)^{-1}$$

LML Components in this case

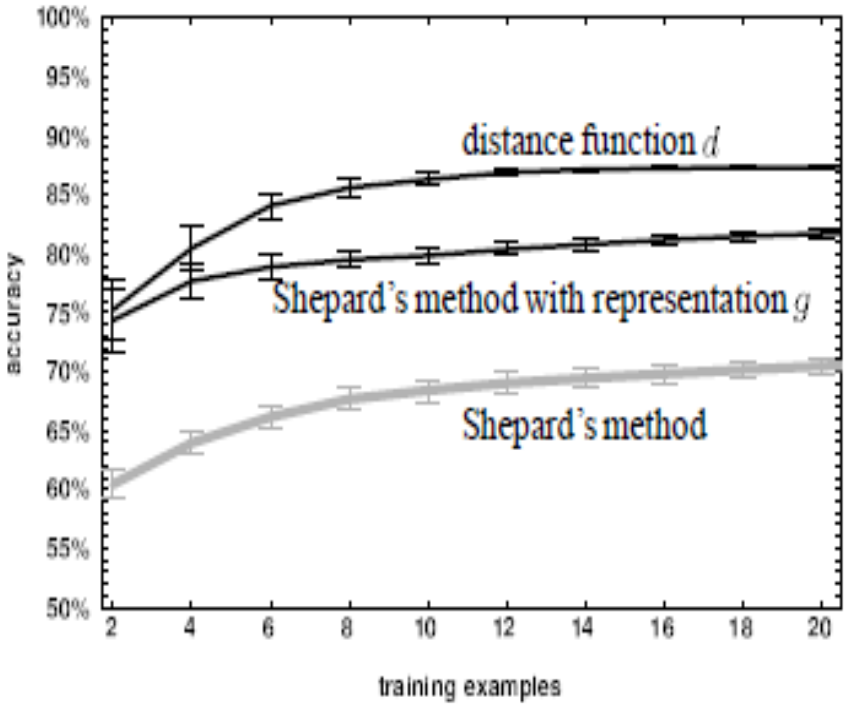
- **PIS**: store all the support sets.
- **KB:** Distance function *d*(x, x'): the probability of example x and x' being the same concept.
- **KM**: Neural network with Back-Propagation.
- **KBL**: The decision making procedure in the last slide.

Neural Network approaches

- Approach 1: based on that in (Caruana, 1993, 1997), which is actually a batch multitask learning approach.
  - simultaneously minimize the error on both the support sets $\{X_k\}$ and the training set X_n
- Approach 2: an *explanation-based neural network (EBNN)*



Results



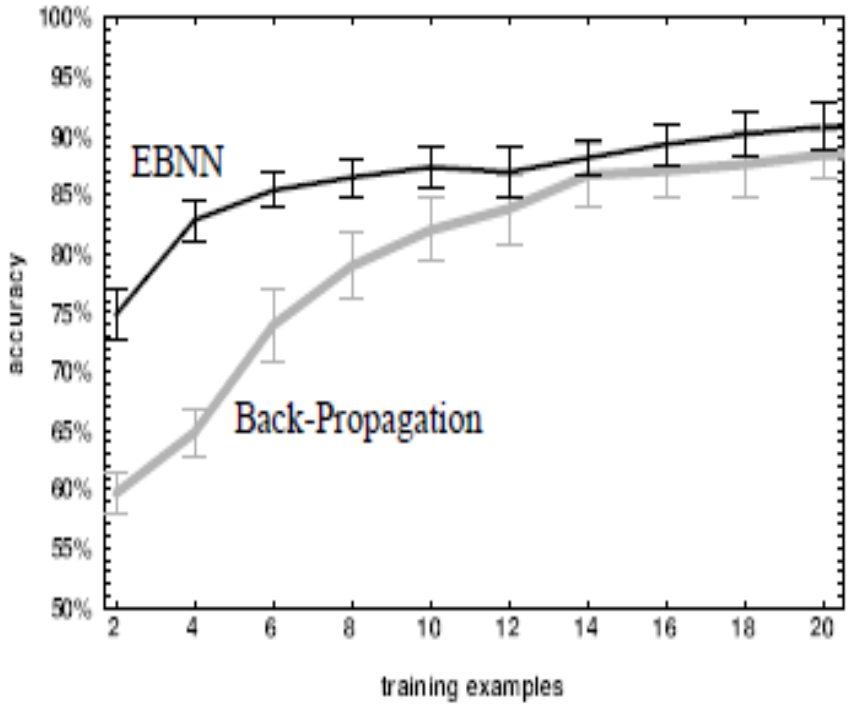


Figure 2: Generalization accuracy as a function of training examples, measured on an independent test set and averaged over 100 experiments. 95%-confidence bars are also displayed.

Task Clustering (TC)

(Thrun and O'Sullivan, 1996)

- In general, not all previous N-1 tasks are similar to the Nth (new) task.
- Based on a similar idea to the lifelong memory-based methods in (Thrun, 1996b),
  - It clusters previous tasks into groups or clusters,
- When the (new) Nth task arrives, it first
  - selects the most similar cluster and then
  - uses the distance function of the cluster for classification in the Nth task.

Some Other Early work on LML

- Constructive inductive learning to deal with learning problem when the original representation space is inadequate for the problem at hand (Michalski, 1993).
- Incremental learning primed on a small, in- complete set of primitive concepts (Solomonoff, 1989)
- Explanation-based neural networks MTL (Thrun, 1996a)
- MTL method of functional (parallel) transfer (Silver & Mercer, 1996)
- Lifelong reinforcement learning method (Tanaka & Yamamura, 1997)
- Collaborative interface agents (Metral & Maes, 1998)

ELLA

(Ruvolo & Eaton, 2013a)

- ELLA: Efficient Lifelong Learning Algorithm
- It is based on GO-MTL (Kumar et al., 2012)
  - A batch multitask learning method
- ELLA is online multitask learning method
  - ELLA is more efficient and can handle a large number of tasks
  - Become a lifelong learning method
    - The model for a new task can be added efficiently.
    - The model for each past task can be updated rapidly.

Inefficiency of GO-MTL

- Since GO-MTL is a batch multitask learning method, the optimization goes through all tasks and their training instances (Kumar et al., 2012).

$$\sum_{t=1}^{T}\sum_{i=1}^{n_t}\mathcal{L}\left(f(\boldsymbol{x}_i^{(t)};\boldsymbol{L}\boldsymbol{s}^{(t)}),y_i^{(t)}\right)+\mu\|\boldsymbol{S}\|_1+\lambda\|\boldsymbol{L}\|_F^2$$

- Very inefficient and impractical for a large number of tasks.
  - It cannot incrementally add a new task efficiently

Initial Objective Function of ELLA

- Objective Function (**Average** rather than sum)

$$e_T(\mathbf{L}) = \frac{1}{T}\sum_{t=1}^{T}\min_{\mathbf{s}^{(t)}}\left\{\frac{1}{n_t}\sum_{i=1}^{n_t}\mathcal{L}\left(f\left(\mathbf{x}_i^{(t)};\mathbf{L}\mathbf{s}^{(t)}\right),y_i^{(t)}\right) + \mu\|\mathbf{s}^{(t)}\|_1\right\} + \lambda\|\mathbf{L}\|_{\mathrm{F}}^2 \ , \qquad (1)$$

Here, $\mathcal{L}$ is the empirical loss function. $(\boldsymbol{x}_i^{(t)}, y_i^{(t)})$ is the ith labeled instance in the training data for task t. $\|\cdot\|_1$ is the L_1 norm, which is controlled by μ as a convex approximation to the true vector sparsity. $\|\boldsymbol{L}\|_F^2$ is the Frobenius norm of matrix $\boldsymbol{L}$ and λ is the regularization coefficient for matrix $\boldsymbol{L}$.

Approximate Equation (1)

- Eliminate the dependence on all of the past training data through the inner summation
  - By using the second-order Taylor expansion of

$$\frac{1}{n_t}\sum_{i=1}^{n_t}\mathcal{L}\left(f\left(\mathbf{x}_i^{(t)};\theta\right),y_i^{(t)}\right)$$

    around $\theta = \theta^{(t)}$ where

$$\theta^{(t)} = \arg\min_\theta \frac{1}{n_t}\sum_{i=1}^{n_t}\mathcal{L}\left(f\left(\mathbf{x}_i^{(t)};\theta,y_i^{(t)}\right)\right)$$

  - is an optimal predictor learned on only the training data on task *t*.

Simplify Optimization

- GO-MTL: when computing a single candidate $\boldsymbol{L}$, an optimization problem must be solved to re-compute the value of each $s^{(t)}$.
- ELLA: after $s^{(t)}$ is computed given the training data for task t, it will not be updated when training on other tasks. Only $\boldsymbol{L}$ will be changed.

- Note: (Ruvolo and Eaton, 2013b) added the mechanism to actively select the next task for learning.

ELLA Accuracy Result

Table 1. The accuracy of ELLA, OMTL, and STL relative to batch multi-task learning (GO-MTL), showing that ELLA achieves nearly equal accuracy to batch MTL and better accuracy than OMTL. The N/A's indicate that OMTL does not handle regression problems. The standard deviation of a value is given after the $\pm$ symbol.

Dataset	Problem Type	Batch MTL Accuracy	ELLA Relative Accuracy	OMTL Relative Accuracy	STL Relative Accuracy
Land Mine	Classification	0.7802 ± 0.013 (AUC)	$99.73 \pm 0.7\%$	$82.2 \pm 3.0\%$	$97.97 \pm 1.5\%$
Facial Expr.	Classification	0.6577 ± 0.021 (AUC)	$99.37 \pm 3.1\%$	$97.58 \pm 3.8\%$	$97.34 \pm 3.9\%$
Syn. Data	Regression	-1.084 ± 0.006 (-rMSE)	$97.74 \pm 2.7\%$	N/A	$92.91 \pm 1.5\%$
London Sch.	Regression	-10.10 ± 0.066 (-rMSE)	$98.90 \pm 1.5\%$	N/A	$97.20 \pm 0.4\%$

ELLA Speed Result

Table 2. The running time of ELLA, OMTL, and STL as compared to batch multi-task learning (GO-MTL), showing that ELLA achieves three orders of magnitude speedup in learning all tasks, and four-to-five orders of magnitude speedup in learning each consecutive new task. The N/A's indicate that OMTL does not handle regression. Speedup was measured relative to the batch method using optimized implementations. The standard deviation of a value is given after the ±.

Dataset	Batch Runtime (seconds)	ELLA All Tasks (speedup)	ELLA New Task (speedup)	OMTL All Tasks (speedup)	OMTL New Task (speedup)	STL All Tasks (speedup)	STL New Task (speedup)
Land Mine	231±6.2	1,350±58	39,150±1,682	22±0.88	638±25	3,342±409	96,918±11,861
Facial Expr.	2,200±92	1,828±100	38,400±2,100	948±65	19,900±1,360	8,511±1,107	178,719±23,239
Syn. Data	1,300±141	5,026±685	502,600±68,500	N/A	N/A	156,489±17,564	1.6E6±1.8E5
London Sch.	715±36	2,721±225	378,219±31,275	N/A	N/A	36,000±4,800	5.0E6±6.7E5

GO-MTL and ELLA in LML

- **PIS**: Stores all the task data
- **KB**: matrix $\boldsymbol{L}$ for K basis tasks and S
- **KM**: optimization (e.g. alternating optimization strategy)
- **KBL**: Each task parameter vector is a linear combination of **KB**, i.e., $\boldsymbol{\theta}^{(t)} = \boldsymbol{L}\boldsymbol{s}^{(t)}$

Lifelong Sentiment Classification

(Chen, Ma, and Liu 2015)

- "*I bought an iPhone a few days ago. It is such a nice phone. The touch screen is really cool. The voice quality is great too. ....*"
- Goal: classify docs or sentences as + or -.
  - Need to manually label a lot of training data for each domain, which is highly labor-intensive
- Can we not label for every domain or at least not label so many docs/sentences?

A Simple Lifelong Learning Method

Assuming we have worked on a *large number of past domains* with all their training data D.

- Build a classifier using D, test on new domain
  - Note - using only one past/source domain as in ***transfer learning*** is not good.
- In many cases – improve accuracy by as much as 19% (= 80%-61%). Why?
- In some others cases – not so good, e.g., it works poorly for toy reviews. Why? "toy"

Lifelong Sentiment Classification

(Chen, Ma and Liu, 2015)

- We need a general solution
- (Chen, Ma and Liu, 2015) adopts a Bayesian optimization framework for LML using stochastic gradient decent
- Lifelong learning uses
  - Word counts from the past data as priors.
  - penalty terms to embed the knowledge gained in the past to deal with domain dependent sentiment words and reliability of knowledge.

Lifelong Learning Components

Past Information Store (PIS): In this work, we do not store the original data used in the past learning tasks, but only their results. For each past learning task $\hat{t}$, we store 1) $P^{\hat{t}}(w|+)$ and $P^{\hat{t}}(w|-)$ for each word w which are from task $\hat{t}$'s NB classifier (see Eq 1); and 2) the number of times that w appears in a positive (+) document $N^{\hat{t}}_{+,w}$ and the number of times that w appears in a negative documents $N^{\hat{t}}_{-,w}$.

Lifelong Learning Components (contd)

Knowledge Base (KB): Our knowledge base contains two types of knowledge:

(a) Document-level knowledge $N^{KB}_{+,w}$ (and $N^{KB}_{-,w}$): number of occurrences of w in the documents of the positive (and negative) class in the past tasks, i.e., $N^{KB}_{+,w} = \sum_{\hat{t}} N^{\hat{t}}_{+,w}$ and $N^{KB}_{-,w} = \sum_{\hat{t}} N^{\hat{t}}_{-,w}$.

(b) Domain-level knowledge $M^{KB}_{+,w}$ (and $M^{KB}_{-,w}$): number of past tasks in which $P(w|+) > P(w|-)$ (and $P(w|+) < P(w|-)$).

Lifelong Learning Components (contd)

Knowledge Miner (KM). Knowledge miner is straightforward as it just performs counting and aggregation of information in PIS to generate knowledge (see (a) and (b) above)

Knowledge-Based Learner (KBL): This learner incorporates knowledge using regularization as penalty terms in our optimization.

Exploiting Knowledge via Penalties

- Handling domain dependent sentiment words

$$\frac{1}{2}\alpha \sum_{w \in V_T} \left(\left(X_{+,w} - N^t_{+,w}\right)^2 + \left(X_{-,w} - N^t_{-,w}\right)^2 \right)$$

- Using domain-level knowledge: If a word appears in one/two past domains/tasks, the knowledge associated with it is probably not reliable or general.

$$\frac{1}{2}\alpha \sum_{w \in V_S} \left(X_{+,w} - R_w \times X^0_{+,w}\right)^2$$

$$+ \frac{1}{2}\alpha \sum_{w \in V_S} \left(X_{-,w} - (1 - R_w) \times X^0_{-,w}\right)^2$$

One Result

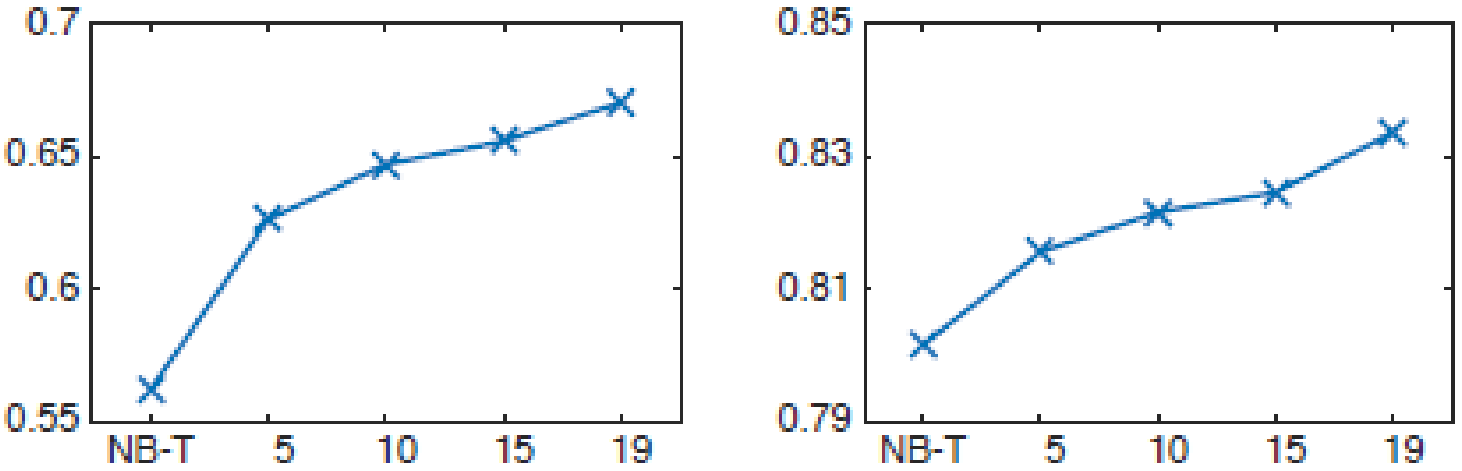


Figure 1: Figure 1. (Left): Negative class F1-score of LSC with #past domains in natural class distribution. (Right): Accuracy of LSC with #past domains in balanced class distribution.

Outline

- Introduction
- A motivating example
- What is lifelong learning?
- Transfer learning
- Multi-task learning
- Supervised lifelong learning
- Semi-supervised never-ending learning
- Unsupervised lifelong topic modeling
- Summary

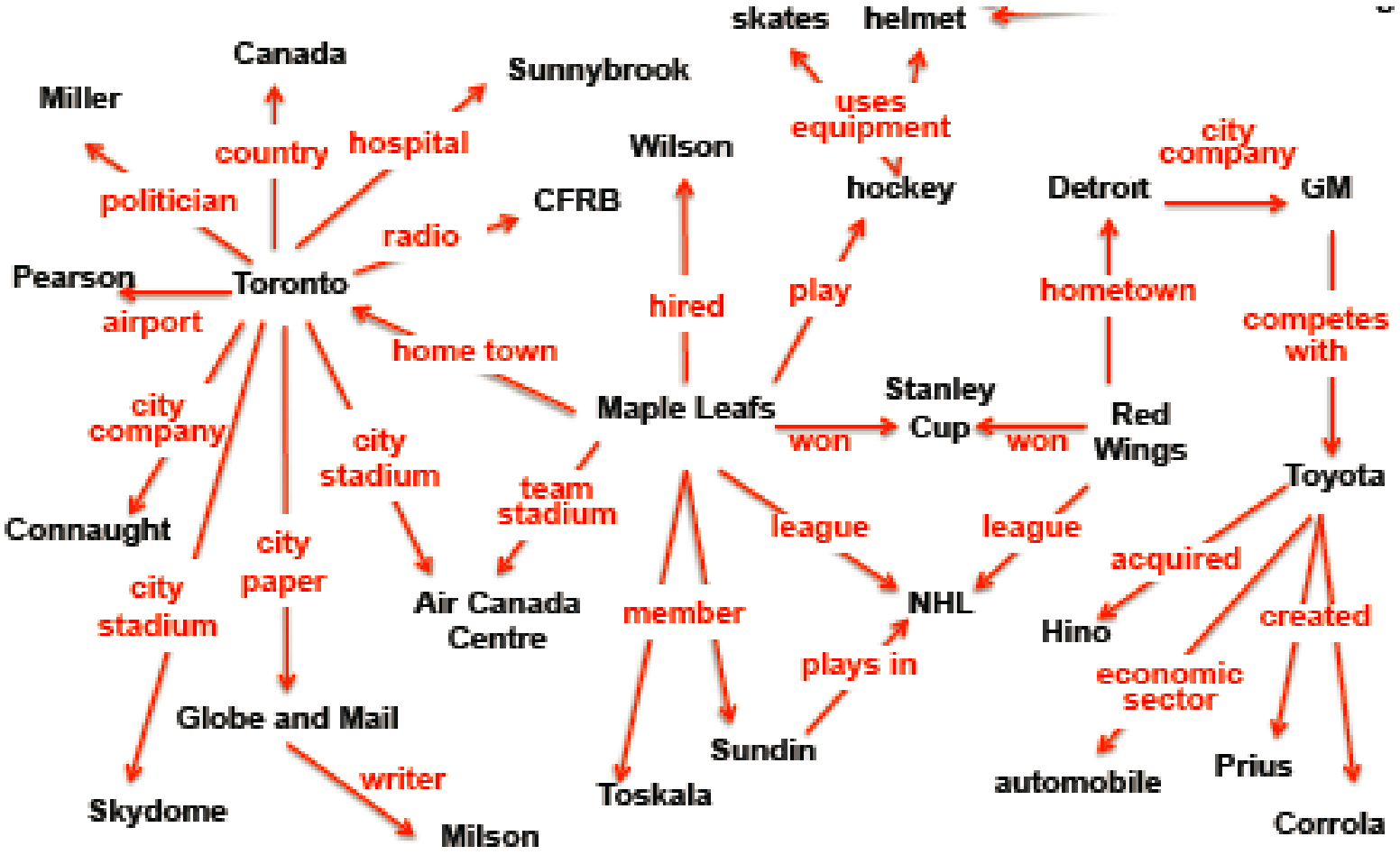
Never Ending Language Learner

(Carlson et al., 2010; Mitchell et al., 2015)

The NELL system:

- Reading task: read web text to extract information to populate a knowledge base of structured facts and knowledge.
- Learning task: learn to read better each day than the day before, as evidenced by its ability to go back to yesterday's text sources and extract more information more accurately.

NELL Knowledge Fragment



LML Components

- PIS in NELL
  - Crawled Web pages
  - Extracted candidate facts from the web text
- KB
  - Consolidate structured facts
- KM
  - A set of classifiers to identify confident facts
- KBL
  - A set of extractors

More about KB

- Instance of category: which noun phrases refer to which specified semantic categories For example, *Los Angeles* is in the category *city*.
- Relationship of a pair of noun phrase, e.g., given a name of an organization and the location, check if *hasOfficesIn(organization, location)*.
- …

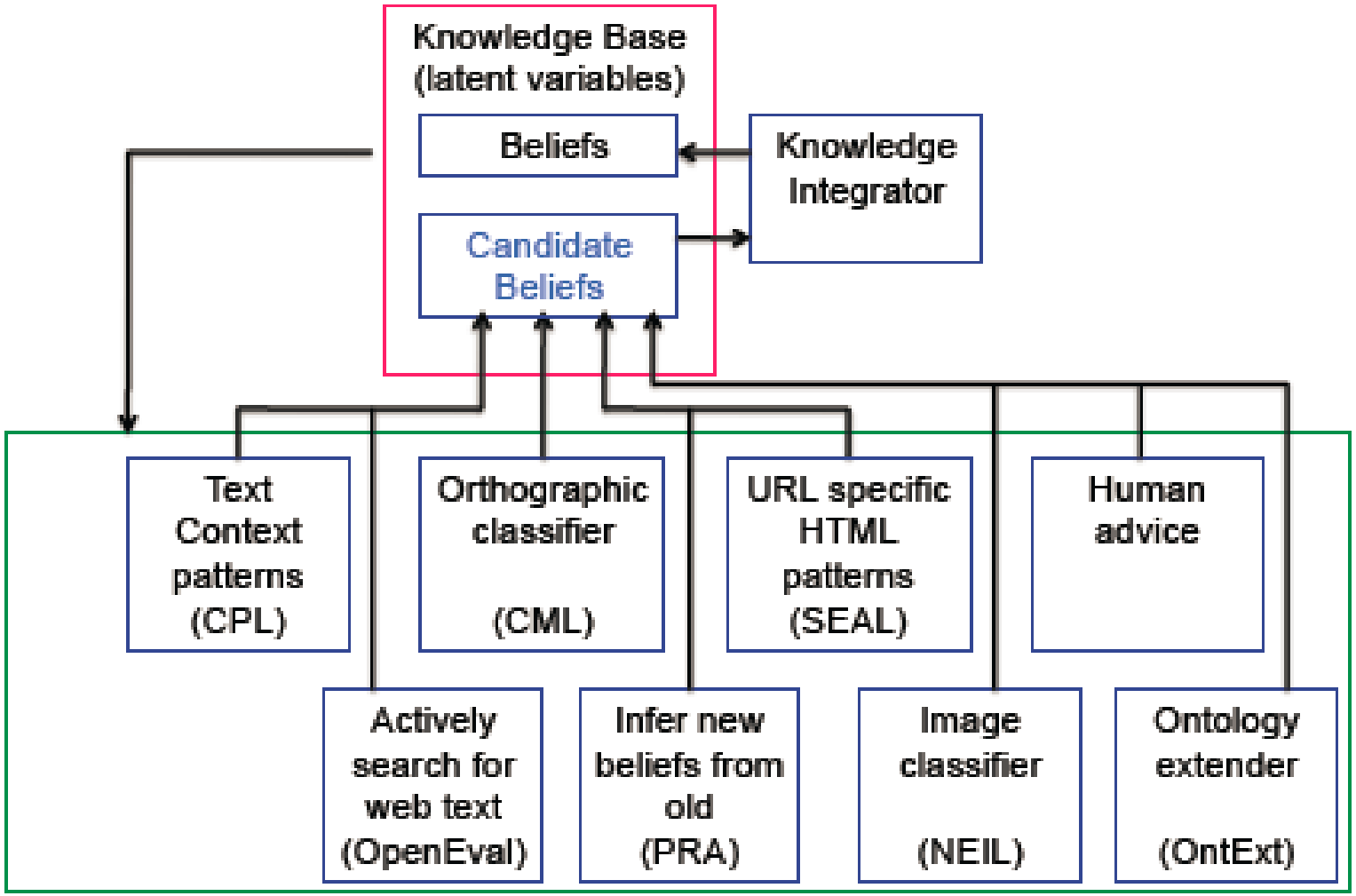
More about KM

- Given identified candidate facts, using classifiers to identify likely correct facts.
  - Classifiers – semi-supervised (manual+self label)
  - employ a threshold to filter those candidates with low-confidence.
  - If a piece of knowledge is validated from multiple sources, promoted even if its confidence is low.
- A first-order learning is also applied to learn probabilistic Horn clauses, which are used to infer new relation instances

KBL in NELL

- Several extractors are used generate candidate facts based on existing knowledge in knowledge base (KB), e.g.,
  - syntactic patterns for identifying entities, categories, and their relationships, such as “X plays for Y,” X scored a goal for Y").
  - lists and tables on webpages for extracting new instances of predicate.
  - …

NELL Architecture



ALICE: Lifelong Info. Extraction

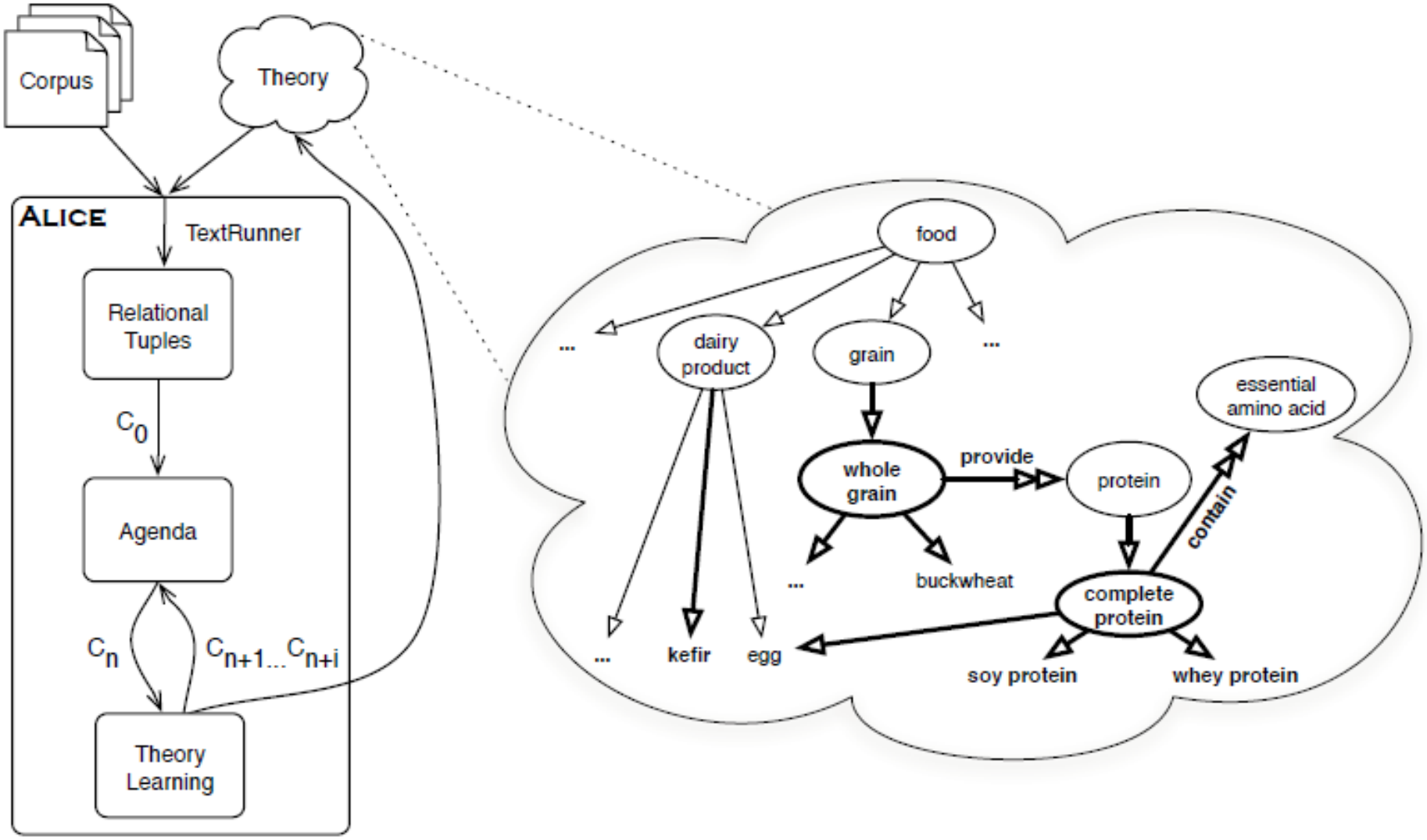
(Banko and Etzioni 2007)

- Similar to NELL, Alice performs similar continuous/lifelong information extraction of
  - concepts and their instances,
  - attributes of concepts, and
  - various relationships among them.
  - The knowledge is iteratively updated
- The extraction also is based on syntactic patterns like
  - (<x> *such as* <y>) and (*fruit such as* <y>),

Lifelong Strategy

- The output knowledge upon completion of a learning task is used in two ways:
  - to update the current domain theory (i.e., domain concept hierarchy and abstraction) and
  - to generate subsequent learning tasks.
- This behavior makes Alice a lifelong agent
  - i.e., Alice uses the knowledge acquired during the *nth* learning task to specify its future learning agenda.
  - Like bootstrapping.

Alice System



Outline

- Introduction
- A motivating example
- What is lifelong learning?
- Transfer learning
- Multi-task learning
- Supervised lifelong learning
- Semi-supervised never-ending learning
- Unsupervised lifelong topic modeling
- Summary

LTM: Lifelong Topic Modeling

(Chen and Liu, ICML-2014)

- Top modeling (Blei et al 2003) find topics from a collection of documents.
  - A document is a distribution over topics
  - A topic is a distribution over terms/words, e.g.,
    - {*price*, *cost*, *cheap*, *expensive*, …}
- **Question**: how to find good past knowledge and use it to help new topic modeling tasks?
- **Data**: product reviews in the sentiment analysis context

Sentiment Analysis (SA) Context

- "*The size is great, but pictures are poor*."
  - Aspects (product features): size, picture
- Why using SA for lifelong learning?
  - Online reviews: Excellent data with extensive sharing of aspect/concepts across domains
    - A large volume for all kinds of products
- Why big (and diverse) data?
  - Learn a broad range of reliable knowledge. More knowledge makes future learning easier.

Key Observation in Practice

- A fair amount of aspect overlapping across reviews of different products or domains
  - Every product review domain has the aspect *price*,
  - Most electronic products share the aspect *battery*
  - Many also share the aspect of *screen*.
- This sharing of concepts / knowledge across domains is true in general, not just for SA.
  - It is rather "silly" not to exploit such sharing in learning

Problem Statement

- Given **a large set of** document collections (big data), $D = \{D_1, ..., D_n\}$, learn from each D_i to produce the result S_i. Let S = U S_i
  - S is called the *topic base*
- **Goal**: Given a test/new collection D^t, learn from D^t with the help of S (and possibly D).
  - $D^t \in D$ or $D^t \notin D$.
  - The results learned this way should be better than without the guidance of S (and D).

Lifelong Learning components

- Past information store (**PIS**): It stores topics/aspects generated in the past tasks.
  - Also called topic base.
- Knowledge base (**KB**): It contains knowledge mined from PIS, dynamically generated must-links
- Knowledge miner (**KM**): Frequent pattern mining using past topics/aspects as transactions.
- Knowledge-based learner (**KBL**): LTM is based on Generalized Pólya Urn Model

What knowledge?

- Should be in the same aspect/topic

  => Must-Links

  e.g., {picture, photo}

- Should not be in the same aspect/topic

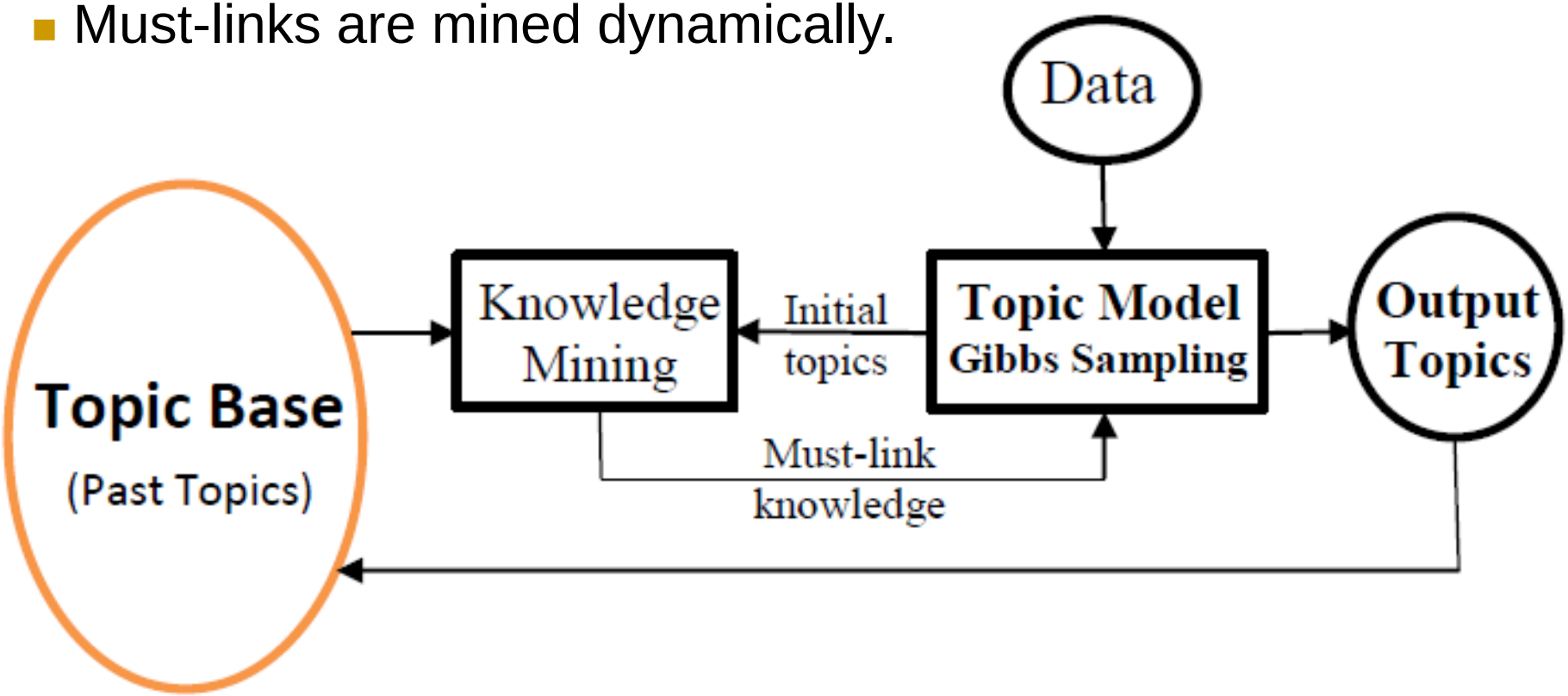
  => Cannot-Links

  e.g., {battery, picture}

Lifelong Topic Modeling (LTM)

(Chen and Liu, ICML-2014)

- Must-links are mined dynamically.



LTM Model

- Step 1: Runs a topic model (e.g., LDA) on each $D_i \in D$ to produce a set of topics S_i called *p-topics*.
- Step 2: (1) Mine prior knowledge (*must-links*) (2) use prior knowledge to guide modeling.

Algorithm 2 LTM(D^t, S)

1: $A^t \leftarrow$ GibbsSampling(D^t, $\emptyset$, N); // Run N Gibbs iterations with no knowledge (equivalent to LDA).
2: **for** $i = 1$ **to** N **do**
3: $K^t \leftarrow$ KnowledgeMining(A^t, S);
4: $A^t \leftarrow$ GibbsSampling(D^t, K^t, 1); // Run with knowledge K^t.
5: **end for**

Knowledge Mining Function

- Topic match: find similar topics (M_{j*}^t) from p-topics for each current topic
- Pattern mining: find frequent itemsets from M_{j*}^t

Algorithm 3 KnowledgeMining(A^t, S)

1: **for** each p-topic $s_k \in S$ **do**
2: $j^* = \min_j$ KL-Divergence(a_j, s_k) for $a_j \in A^t$;
3: **if** KL-Divergence(a_{j^*}, s_k) $\leq \pi$ **then**
4: $M_{j*}^t \leftarrow M_{j*}^t \cup s_k$;
5: **end if**
6: **end for**
7: $K^t \leftarrow \cup_{j^*}$ FIM(M_{j*}^t); // Frequent Itermset Mining.

An Example

- Given a newly discovered topic:

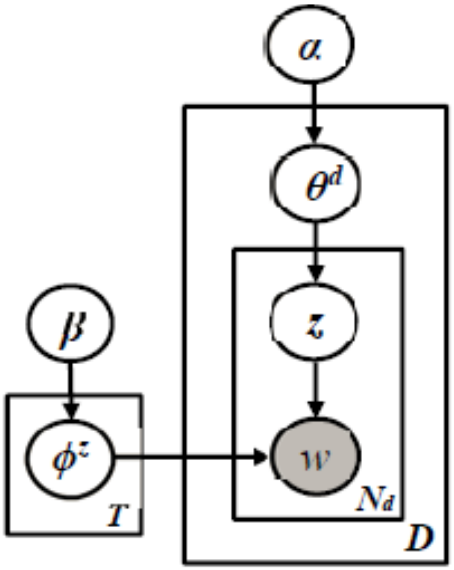
  {*price*, *book*, *cost*, *seller*, *money*},

  - We find 3 matching topics from topic base *S*
    - Domain 1: {*price*, *color*, *cost*, *life*, *picture*}
    - Domain 2: {*cost*, *screen*, *price*, *expensive*, *voice*}
    - Domain 3: {*price*, *money*, *customer*, *service*, *expensive*}

- If we require words appear in at least two domains, we get two must-links (knowledge):
  - {*price*, *cost*} and {*price*, *expensive*}.
  - Each set is likely to belong to the same aspect/topic.

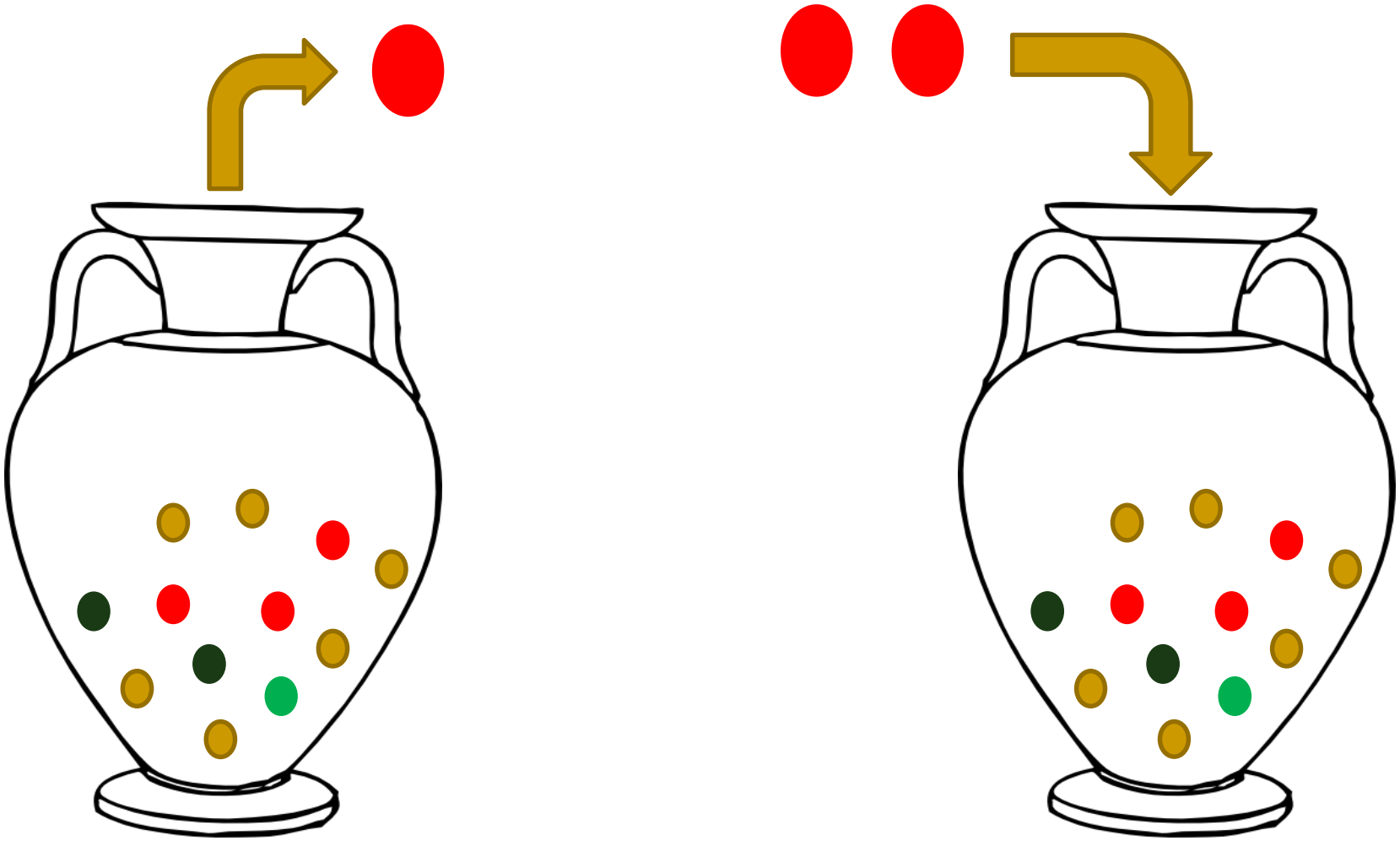
Model Inference: Gibbs Sampling

- How to use the *must-links* knowledge?
  - e.g., {*price*, *cost*} & {*price*, *expensive*}

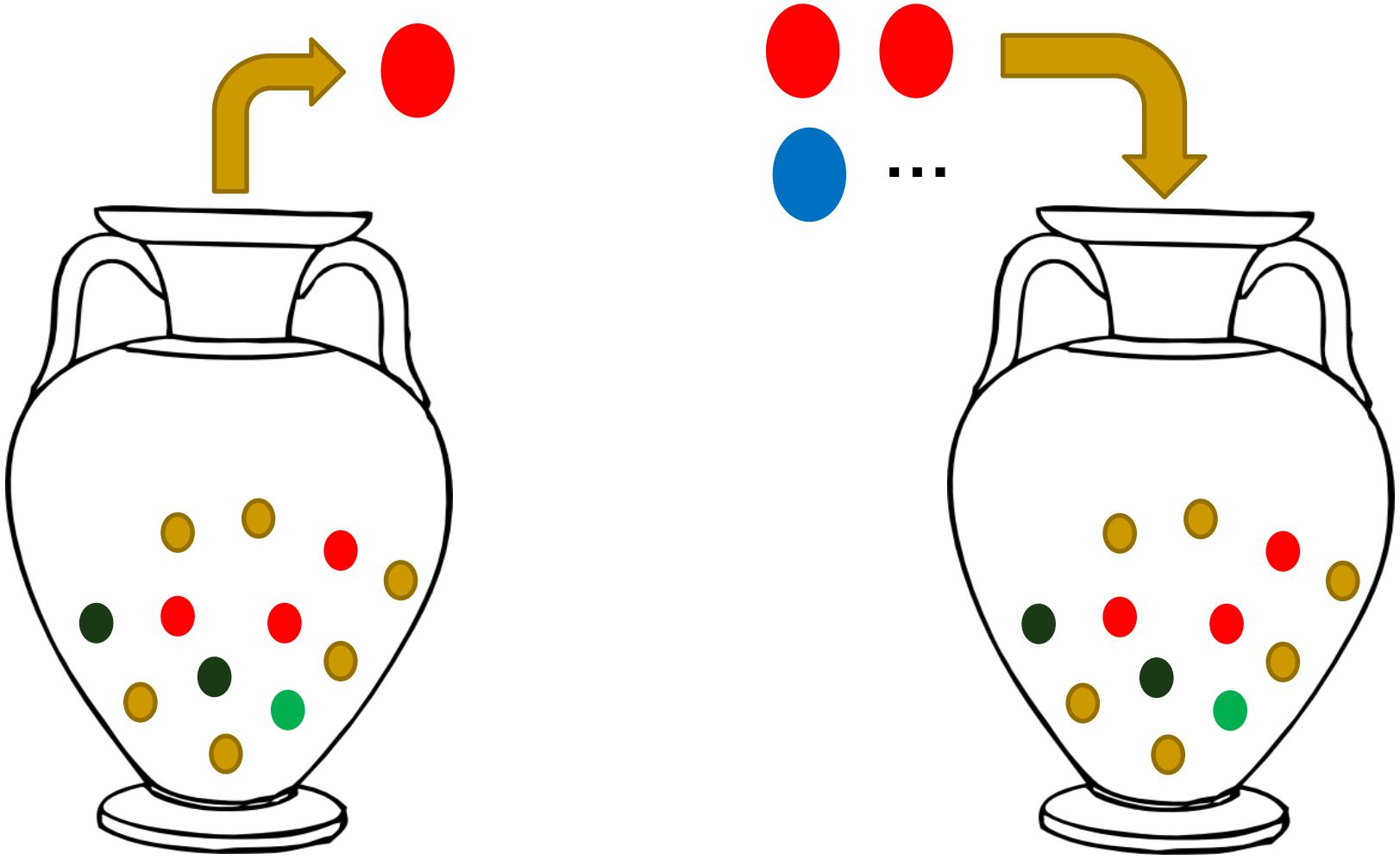


- Graphical model: same as LDA
- But the model inference is very different
  - Generalized Pólya Urn Model (GPU)
- Idea: When assigning a topic *t* to a word *w*, also assign *a fraction of t* to words in must-links sharing with *w*.

Simple Pólya Urn model (SPU)



Generalized Pólya Urn model (GPU)



Gibbs Sampler for GPU

- $P\left(z_i = t \,\middle|\, \boldsymbol{z}^{-i}, \boldsymbol{w}, \alpha, \beta\right) \propto$

$$\frac{n_{m,t}^{-i} + \alpha}{\sum_{t'=1}^{T}\left(n_{m,t'}^{-i} + \alpha\right)} \times \frac{\sum_{w'=1}^{V} \mathbb{A}_{w',w_i} \times n_{t,w'}^{-i} + \beta}{\sum_{v=1}^{V}\left(\sum_{w'=1}^{V} \mathbb{A}_{w',v} \times n_{t,w'}^{-i} + \beta\right)}$$

Experiment Results

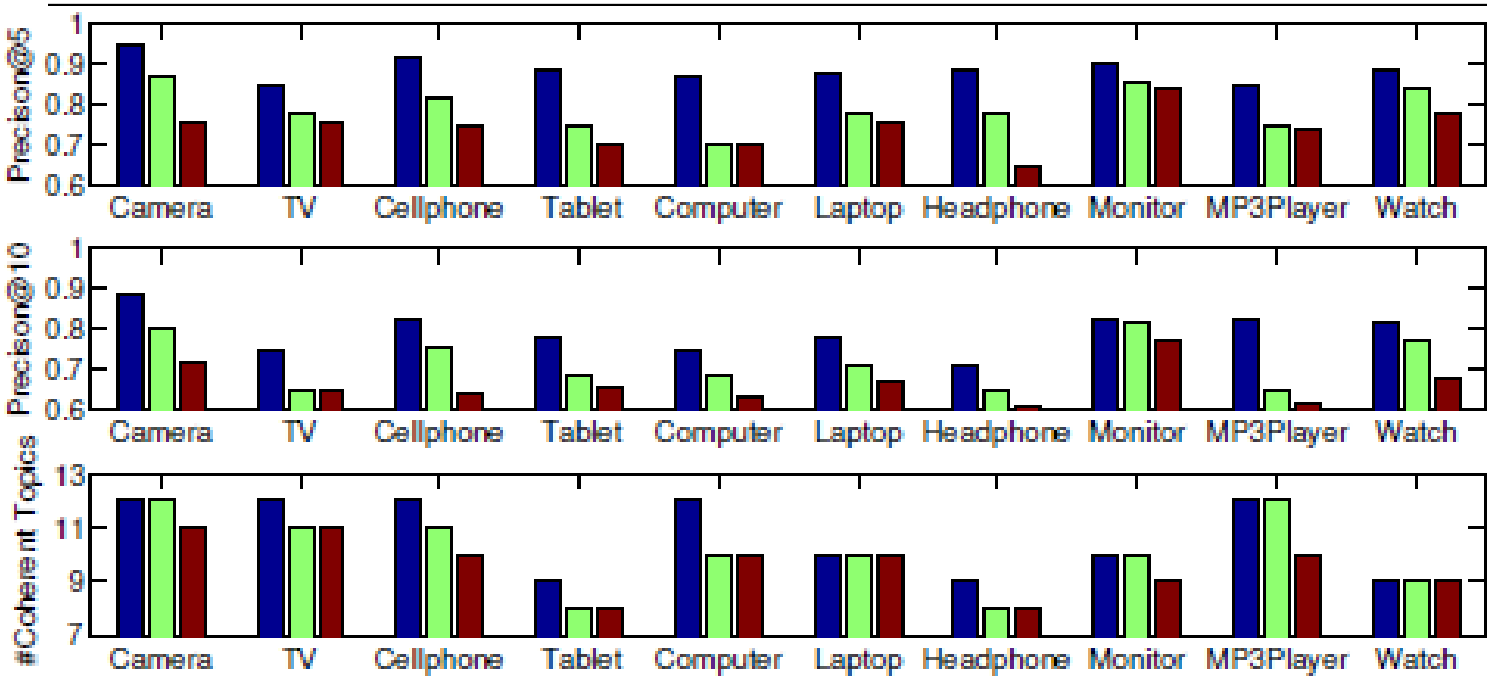


Figure 2. Top & Middle: Topical words *Precision*@5 & *Presicion*@10 of coherent topics of each model respectively; Bottom: number of coherent (#Coherent) topics discovered by each model. The bars from left to right in each group are for LTM, LDA, and DF-LDA. On average, for *Precision*@5 and

AMC: Modeling with Small Datasets

(Chen and Liu, KDD- 2014)

- The LTM model is not sufficient when the data is small for each task because
  - It cannot produce good initial topics for matching to identify relevant past topics.
- AMC mines must-links differently
  - Mine must-links from the past information store without considering the target task/data
    - Task/domain independent.
  - Using FIM to mine from all past topics.

Lifelong Learning components

- Past information store (**PIS**): It stores topics/aspects generated in the past tasks.
  - called topic base.
- Knowledge base (KB): It contains knowledge mined from PIS,
  - must-links generated off-line and cannot-links generated dynamically
- Knowledge miner: Frequent pattern mining & …
- Knowledge-based learner: LTM based on multi-generalized Polya urn Model

Cannot-Links

- In this case, we need to mine cannot-links, which is tricky because
  - There is a huge number of cannot-links $O(V^2)$
    - V is the vocabulary size
- We thus need to focus on only those terms that are relevant to target data D^t.
  - That is, we need to embed the process of finding cannot-links in the sampling

Overall Algorithm

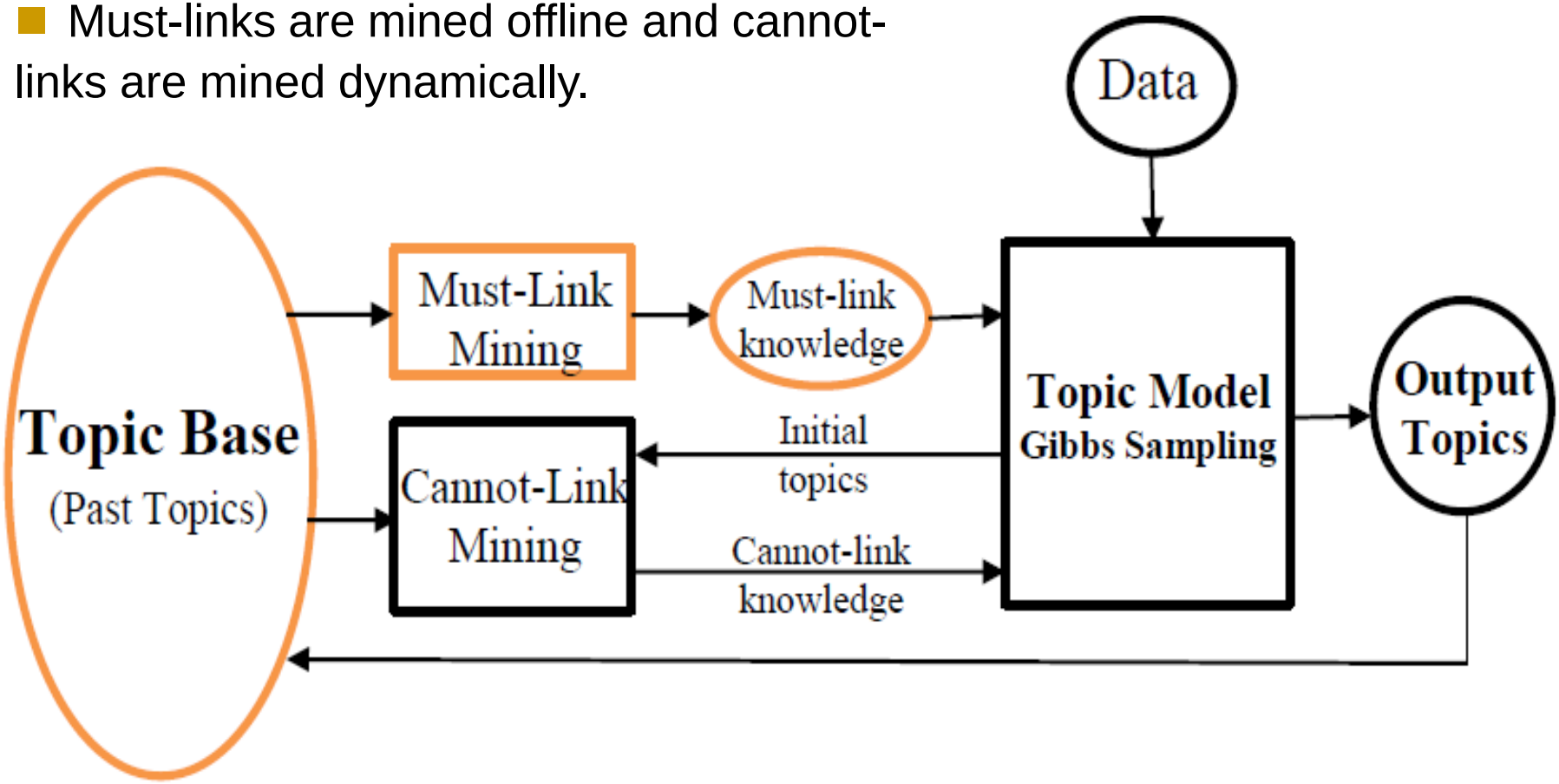
Algorithm 1 AMC(D^t, S, M)

1: $A^t \leftarrow$ GibbsSampling(D^t, N, M, $\emptyset$); // $\emptyset$: no cannot-links.
2: **for** $r = 1$ **to** R **do**
3: $\quad C \leftarrow C \cup$ MineCannotLinks(S, A^t);
4: $\quad A^t \leftarrow$ GibbsSampling(D^t, N, M, C);
5: **end for**
6: $S \leftarrow$ Incorporate(A^t, S);
7: $M \leftarrow$ MiningMustLinks(S);

- Sampling becomes much more complex
  - The paper proposed M-GPU model (multi-generalized Polya urn model)

Lifelong Topic Modeling – AMC

- Must-links are mined offline and cannot-links are mined dynamically.



Reflection on Sentiment Applications

- Sentiment analysis (SA): two key concepts form its core
  - (1) sentiment and (2) sentiment target or aspect
- Key observation: Due to highly focused nature, SA tasks and data have a significant amount of sharing of sentiment and aspect expressions
  - which makes *lifelong learning* promising
- Data: a huge volume of reviews of all kinds
- Unlimited applications

Some Related Unsupervised Work

- Unsupervised ART (Adaptive Resonance Theory) neural networks (Grossberg 1987).
- A cluster ensemble framework, using multiple partitionings of a set objects without accessing the original features (Strehl and Ghosh 2003).
- Self-taught learning, using unlabeled data to construct higher-level features (Raina et al., 2007) for classification.

Outline

- Introduction
- A motivating example
- What is lifelong learning?
- Transfer learning
- Multitask learning
- Supervised lifelong learning
- Semi-supervised never-ending learning
- Unsupervised lifelong topic modeling
- Summary

Summary

- This tutorial gave an introduction to LML.
  - by no means exhaustive, e.g.,
    - reinforcement LML (Ring 1997; Sutton, Koop, and Silver 2007)
    - theory (Pentina and Lampert, 2014)
- Existing LML research is still in its infancy.
  - Most are special cases of LML, e.g., transfer learning and (batch) multitask learning.
  - Our understanding of LML is very limited.
  - Current research mainly focuses on
    - Only one type of tasks in a system

Summary

- Future systems should learn and use mixed types of knowledge in one system
- LML needs big data – to learn a large amount of reliable knowledge of different types.
  - Little knowledge is not very useful
- Big data offers a good opportunity for LML.
  - LML for NLP is particularly promising due to *extensive concept sharing cross domains*
    - same word in different domains has similar meanings

Summary

There are many challenges for LML, e.g.,

- It is desirable to retain as much information and knowledge as possible from the past, but
  - How to “remember” them over time effectively
  - How to represent different forms of knowledge
  - How to consolidate and meta-mine knowledge
  - How to find relevant knowledge to apply.
- What is the general way of using different types of knowledge in learning?

Thank You!

Reference (1)

Abernethy, Jacob, Bartlett, Peter, and Rakhlin, Alexander. Multitask Learning with Expert Advice. In COLT, pp. 484–498, 2007.

Agarwal, Alekh, Rakhlin, Alexander, and Bartlett, Peter. Matrix regularization techniques for online multitask learning. EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2008-138, 2008.

Ammar, Haitham B, Eaton, Eric, Ruvolo, Paul, and Taylor, Matthew. Online Multi-Task Learning for Policy Gradient Methods. In ICML, pp. 1206–1214, 2014.

Ando, Rie Kubota and Zhang, Tong. A High-performance Semi-supervised Learning Method for Text Chunking. In ACL, pp. 1–9, 2005.

Argyriou, Andreas, Evgeniou, Theodoros, and Pontil, Massimiliano. Convex Multi-task Feature Learning. Machine Learning, 73(3):243–272, 2008.

Banko, Michele and Etzioni, Oren. Strategies for Lifelong Knowledge Extraction from the Web. In K-CAP, pp. 95– 102, 2007.

Baxter, Jonathan. A Model of Inductive Bias Learning. Journal of Artificial Intelligence Research, 12:149–198, 2000.

Reference (2)

Ben-David, Shai and Schuller, Reba. Exploiting Task Relatedness for Multiple Task Learning. In COLT, 2003.

Bickel, Steffen, Bru¨ckner, Michael, and Scheffer, Tobias. Discriminative Learning for Differing Training and Test Distributions. In ICML, pp. 81–88, 2007.

Bickel, Steffen, Bogojeska, Jasmina, Lengauer, Thomas, and Scheffer, Tobias. Multi-task Learning for HIV Therapy Screening. In ICML, pp. 56–63, 2008.

Blitzer, John, McDonald, Ryan, and Pereira, Fernando. Domain Adaptation with Structural Correspondence Learning. In EMNLP, pp. 120–128, 2006.

Blitzer, John, Dredze, Mark, and Pereira, Fernando. Biographies, Bollywood, Boom-boxes and Blenders: Domain Adaptation for Sentiment Classification. In ACL, pp. 440–447, 2007.

Bonilla, Edwin V, Chai, Kian M, and Williams, Christopher. Multi-task Gaussian process prediction. In NIPS, pp. 153–160, 2007.

Carlson, Andrew, Betteridge, Justin, and Kisiel, Bryan. Toward an Architecture for Never-Ending Language Learning. In AAAI, pp. 1306–1313, 2010.

Reference (3)

Caruana, Rich. Multitask Learning. Machine learning, 28 (1):41–75, 1997.

Cavallanti, Giovanni, Cesa-Bianchi, Nicolo`, and Gentile, Claudio. Linear Algorithms for Online Multitask Classification. Journal of Machine Learning Research, 11: 2901–2934, 2010.

Chen, Jianhui, Tang, Lei, Liu, Jun, and Ye, Jieping. A Convex Formulation for Learning Shared Structures from Multiple Tasks. In ICML, pp. 137–144, 2009.

Chen, Jianhui, Zhou, Jiayu, and Ye, Jieping. Integrating low-rank and group-sparse structures for robust multitask learning. In KDD, pp. 42–50, 2011.

Chen, Zhiyuan and Liu, Bing. Topic Modeling using Topics from Many Domains, Lifelong Learning and Big Data. In ICML, pp. 703–711, 2014a.

Chen, Zhiyuan and Liu, Bing. Mining Topics in Documents : Standing on the Shoulders of Big Data. In KDD, pp. 1116–1125, 2014b.

Chen, Zhiyuan, Liu, Bing, and Hsu, M. Identifying Intention Posts in Discussion Forums. In NAACL-HLT, number June, pp. 1041–1050, 2013.

Chen, Zhiyuan, Ma, Nianzu, and Liu, Bing. Lifelong Learning for Sentiment Classification. In ACL, 2015.

Reference (4)

Dai, Wenyuan, Xue, Gui-Rong, Yang, Qiang, and Yu, Yong. Co-clustering Based Classification for Out-ofdomain Documents. In KDD, 2007a.

Dai, Wenyuan, Xue, Gui-rong, Yang, Qiang, and Yu, Yong. Transferring naive bayes classifiers for text classification. In AAAI, 2007b.

Dai, Wenyuan, Yang, Qiang, Xue, Gui-Rong, and Yu, Yong. Boosting for Transfer Learning. In ICML, pp. 193–200, 2007c.

Daume III, Hal. Frustratingly Easy Domain Adaptation. In ACL, 2007.

Daume´ III, Hal. Bayesian Multitask Learning with Latent Hierarchies. In UAI, pp. 135–142, 2009.

Dekel, Ofer, Long, Philip M, and Singer, Yoram. Online multitask learning. In COLT, pp. 453–467. Springer, 2006.

Evgeniou, Theodoros and Pontil, Massimiliano. Regularized Multi–task Learning. In KDD, pp. 109–117, 2004.

Gao, Jing, Fan, Wei, Jiang, Jing, and Han, Jiawei. Knowledge Transfer via Multiple Model Local Structure Mapping. In KDD, pp. 283–291, 2008.

Gong, Pinghua, Ye, Jieping, and Zhang, Changshui. Robust Multi-task Feature Learning. In KDD, pp. 895–903, 2012.

Reference (5)

Grossberg, Stephen. Competitive learning: From interactive activation to adaptive resonance. Cognitive science, 11(1):23–63, 1987.

Jacob, Laurent, Vert, Jean-philippe, and Bach, Francis R. Clustered Multi-Task Learning: A Convex Formulation. In NIPS, pp. 745–752. 2009.

Jiang, Jing and Zhai, ChengXiang. Instance weighting for domain adaptation in NLP. In ACL, volume 7, pp. 264– 271, 2007.

Kang, Zhuoliang, Grauman, Kristen, and Sha, Fei. Learning with Whom to Share in Multi-task Feature Learning. In ICML, pp. 521–528, 2011.

Kumar, Abhishek, Daum, Hal, and Iii, Hal Daume. Learning Task Grouping and Overlap in Multi-task Learning. In ICML, pp. 1383–1390, 2012.

Lawrence, Neil D and Platt, John C. Learning to Learn with the Informative Vector Machine. In ICML, 2004.

Lee, Su-In, Chatalbashev, Vassil, Vickrey, David, and Koller, Daphne. Learning a Meta-level Prior for Feature Relevance from Multiple Related Tasks. In ICML, pp. 489–496, 2007.

Li, Cong, Michael Georgiopoulos, and Georgios C Anagnostopoulos. A unifying framework for typical multitask multiple kernel learning problems. Neural Networks and Learning Systems, IEEE Transactions on, 25(7): 2014.

Reference (6)

Liao, Xuejun, Xue, Ya, and Carin, Lawrence. Logistic Regression with an Auxiliary Data Source. In ICML, pp. 505–512, 2005.

Liu, Bing. Sentiment Analysis and Opinion Mining. Synthesis Lectures on Human Language Technologies, 5(1): 1–167, 2012.

Liu, Bing. Sentiment Analysis Mining Opinions, Sentiments, and Emotions. Cambridge University Press, 2015.

Lugosi, Ga´bor, Papaspiliopoulos, Omiros, and Stoltz, Gilles. Online multi-task learning with hard constraints. In COLT, 2009.

Metral, Y Lashkari M and Maes, Pattie. Collaborative interface agents. Readings in agents, pp. 111, 1998.

Michalski, Ryszard S. Learning= inferencing+ memorizing. In Foundations of Knowledge Acquisition, pp. 1–41. Springer, 1993.

Mitchell, T, Cohen, W, Hruschka, E, Talukdar, P, Betteridge, J, Carlson, A, Dalvi, B, Gardner, M, Kisiel, B, Krishnamurthy, J, Lao, N, Mazaitis, K, Mohamed, T, Nakashole, N, Platanios, E, Ritter, A, Samadi, M, Settles, B, Wang, R, Wijaya, D, Gupta, A, Chen, X, Saparov, A, Greaves, M, and Welling, J. Never-Ending Learning. In AAAI, 2015.

Reference (7)

Pan, Sinno Jialin and Yang, Qiang. A Survey on Transfer Learning. IEEE Trans. Knowl. Data Eng., 22(10):1345– 1359, 2010.

Pentina, Anastasia, and Lampert, Christoph H. A PAC-Bayesian Bound for Lifelong Learning. ICML-2014.

Raina, Rajat, Battle, Alexis, Lee, Honglak, Packer, Benjamin, and Ng, Andrew Y. Self-taught Learning : Transfer Learning from Unlabeled Data. In ICML, 2007.

Rigutini, Leonardo, Maggini, Marco, and Liu, Bing. An EM Based Training Algorithm for Cross-Language Text Categorization. In WI, pp. 529–535, 2005.

Ring, Mark B. CHILD: A first step towards continual learning. Machine Learning, 104:77–104, 1997.

Ruvolo, Paul and Eaton, Eric. ELLA: An efficient lifelong learning algorithm. In ICML, pp. 507–515, 2013a.

Ruvolo, Paul and Eaton, Eric. Active Task Selection for Lifelong Machine Learning. In AAAI, pp. 862–868, 2013b.

Ruvolo, Paul and Eaton, Eric. Online multi-task learning via sparse dictionary optimization. In AAAI, 2014.

Reference (8)

Saha, Avishek, Rai, Piyush, Venkatasubramanian, Suresh, and Daume, Hal. Online learning of multiple tasks and their relationships. In AISTATS, 2011.

Schwaighofer, Anton, Tresp, Volker, and Yu, Kai. Learning Gaussian process kernels via hierarchical Bayes. In NIPS, pp. 1209–1216, 2004.

Shultz, Thomas R and Rivest, Francois. Knowledge-based cascade-correlation: Using knowledge to speed learning. Connection Science, 13(1):43–72, 2001.

Strehl, Alexander and Ghosh, Joydeep. Cluster Ensembles — a Knowledge Reuse Framework for Combining Mul- tiple Partitions. Journal of Machine Learning Research, 3:583–617, 2003.

Silver, Daniel L and Mercer, Robert. The parallel transfer of task knowledge using dynamic learning rates based on a measure of relatedness. Connection Science, 8(2): 277–294, 1996.

Silver, Daniel L, Poirier, Ryan, and Currie, Duane. Inductive transfer with context-sensitive neural networks. Machine Learning, 73(3):313–336, 2008.

Silver, Daniel L, Yang, Qiang, and Li, Lianghao. Lifelong Machine Learning Systems: Beyond Learning Algorithms. In AAAI Spring Symposium: Lifelong Machine Learning, pp. 49–55, 2013.

Reference (9)

Solomonoff, Ray J. A system for incremental learning based on algorithmic probability. In Proceedings of the Sixth Israeli Conference on Artificial Intelligence, Computer Vision and Pattern Recognition, pp. 515–527, 1989.

Sugiyama, Masashi, Nakajima, Shinichi, Kashima, Hisashi, Buenau, Paul V, and Kawanabe, Motoaki. Direct importance estimation with model selection and its application to covariate shift adaptation. In NIPS, pp. 1433–1440, 2008.

Sutton, Richard S, Koop, Anna, and Silver, David. On the Role of Tracking in Stationary Environments. In ICML, pp. 871–878, 2007.

Tanaka, Fumihide and Yamamura, Masayuki. An approach to lifelong reinforcement learning through multiple environments. In 6th European Workshop on Learning Robots, pp. 93–99, 1997.

Thrun, Sebastian. Explanation-Based Neural Network Learning: A Lifelong Learning Approach. Kluwer Academic Publishers, 1996a.

Thrun, Sebastian. Is learning the n-th thing any easier than learning the first? In NIPS, pp. 640–646, 1996b.

Thrun, Sebastian and O'Sullivan, Joseph. Discovering Structure in Multiple Learning Tasks: The TC Algorithm. In ICML, pp. 489–497. 1996.

Reference (10)

Wang, Chang and Mahadevan, Sridhar. Manifold Alignment Using Procrustes Analysis. In ICML, pp. 1120– 1127, 2008.

Xue, Ya, Liao, Xuejun, Carin, Lawrence, and Krishnapuram, Balaji. Multi-Task Learning for Classification with Dirichlet Process Priors. Journal of Machine Learning Research, 8:35–63, 2007.

Yu, Kai, Tresp, Volker, and Schwaighofer, Anton. Learning Gaussian Processes from Multiple Tasks. In ICML, pp. 1012–1019, 2005.

Yu, Shipeng, Tresp, Volker, and Yu, Kai. Robust Multi-task Learning with T-processes. In ICML, pp. 1103–1110, 2007.

Yuan, Xiao-Tong, Liu, Xiaobai, and Yan, Shuicheng. Visual classification with multitask joint sparse representation. Image Processing, IEEE Transactions on, 21(10): 4349–4360, 2012.

Zhou, Jiayu, Yuan, Lei, Liu, Jun, and Ye, Jieping. A multitask learning formulation for predicting disease progression. In KDD, pp. 814–822, 2011.

Zhu, Jun, Chen, Ning, and Xing, Eric P. Infinite latent SVM for classification and multi-task learning. In NIPS, pp. 1620–1628, 2011.