Lifelong Machine Learning and Computer Reading the Web

Zhiyuan (Brett) Chen, Google
Estevam Hruschka, UFSCar, CMU
Bing Liu, University of Illinois at Chicago
Introduction

- Classic Machine Learning (ML) paradigm (isolated single-task learning)
  - Given a dataset, run a ML algo. to build a model
  - Without considering the past learned knowledge

- 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
  - Isolated learning has limitations.
Introduction: ML 1.0 limitation

- Learned knowledge is not cumulative
- **No memory**: Knowledge learned isn’t retained
  - ML cannot learn by leveraging the past knowledge
- Due to the lack of prior knowledge
  - ML needs a large number of training examples.
- **Without knowledge accumulation and self-learning (with no supervision)**
  - It is impossible to build a truly intelligent system
    - Cannot imagine that for every task a large number of training examples need to be labeled by humans
Introduction: human learning

- Humans never learn in isolation
- We learn effectively from a few examples with the help of the past knowledge.
  - Nobody has ever given me 1000 positive and 1000 negative docs, and asked me to build a classifier manually
- Whenever we see a new situation, a large part of it is known to us. Little is completely new!
Introduction: ML 2.0
(Thrun, 1996b; Silver et al 2013; Chen and Liu, 2014a)

- **Lifelong Machine Learning (LML)**
  - Learn as humans do
  - Retain learned knowledge from previous tasks & use it to help future learning

- Let us call this paradigm **Machine Learning 2.0**
  - LML may require a systems approach
  - Multiple tasks with multiple learning/mining algorithms
Introduction: LML with Big Data

- Big data provides a great opportunity for LML
  - Abundant information from the Web
  - Extensive sharing of concepts across tasks/domains
  - Example: natural language learning tasks on different sources are all related
Outline

- A motivating example
- What is lifelong machine learning?
- Related learning tasks
- Lifelong supervised learning
- Semi-supervised never-ending learning
- Lifelong unsupervised learning
- Lifelong reinforcement learning
- Summary
Outline

- A motivating example
- What is lifelong machine learning?
- Related learning tasks
- Lifelong supervised learning
- Semi-supervised never-ending learning
- Lifelong unsupervised learning
- Lifelong reinforcement learning
- 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 with unlimited applications
  - Useful to every organization and individual
  - Example: online shopping
A Motivating Example
(Liu, 2012; 2015)

- Sentiment analysis is 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 but the battery dies fast.”
(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 source domain (camera) to help learning in the target domain (earphone)
  - Two domains need to be very similar

- 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 a 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”
“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.
Outline

- A motivating example
- What is lifelong machine learning?
- Related learning tasks
- Lifelong supervised learning
- Semi-supervised never-ending learning
- Lifelong unsupervised learning
- Lifelong reinforcement learning
- Summary
Lifelong Machine Learning (LML)

Definition: LML is a continuous learning process where the learner has performed a sequence of $N$ learning tasks, $T_1$, $T_2$, ..., $T_N$.

- When faced with the $N$th task $T_{N+1}$ with its data $D_{N+1}$, the learner makes use of the prior knowledge $K$ in its knowledge base (KB) to help learn $T_{N+1}$.
- KB contains all the knowledge accumulated in the past learning of the $N$ tasks.
- After learning $T_{N+1}$, KB is updated with the learned (intermediate as well the final) results from $T_{N+1}$. 
Key Characteristics of LML

- Continuous learning process
- Knowledge accumulation in KB
- Use of past knowledge to help future learning
Components of LML

- Knowledge Base (KB)
  - Past Information Store (PIS)
  - Knowledge Store (KS)
  - Knowledge Miner (KM)
  - Knowledge Reasoner (KR)

- 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 Store (KS)

- 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 KS (Knowledge Store)
Knowledge Reasoner (KR)

- It makes inference in the KB to generate additional knowledge.
- Most current LML systems do not have this capability.
- However, with the advance of LML, this component will become important.
Knowledge-Based Learner (KBL)

- Given the knowledge in KS, 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 \((N+1)\)th task by using knowledge gained from the past \(N\) tasks.
- It can also improve any of the models from the past \(N\) tasks based on results from the other \(N\) tasks (including the \((N+1)\)th task):
  - By treating that previous task as the “\((N+1)\)th” task.
Outline

- A motivating example
- What is lifelong machine learning?
- Related learning tasks
  - Lifelong supervised learning
  - Semi-supervised never-ending learning
  - Lifelong unsupervised learning
  - Lifelong reinforcement learning
- Summary
Transfer learning

- **Source domain(s):** With labeled training data
- **Target domain:** With little/no labeled training data
- **Goal:** leverage the information from the source domain(s) to help learning in the target domain
  - Only optimize the target domain/task learning
A Large Body of Literature

- Transfer learning has been a popular research topic and researched in many fields, e.g.,
  - Machine learning
  - Data mining
  - Natural language processing
  - Computer vision
- Pan & Yang (2010) presented an excellent survey with extensive references.
One Transfer Learning Technique

- Structural correspondence learning (SCL) (Blitzer et al 2006)

- Pivot features
  - Have the same characteristics or behaviors in both domains
  - Non-pivot features which are correlated with many of the same pivot features are assumed to correspond
Choosing Pivot Features

- For different applications, pivot features may be chosen differently, for example,
  - For part-of-speech tagging, frequently-occurring words in both domains are good choices (Blitzer et al., 2006)
  - For sentiment classification, pivot 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

\[ f_\ell(x) = \text{sgn}(\hat{w}_\ell \cdot x), \quad \ell = 1 \ldots m \]

- Using unlabeled data (predicting whether the pivot feature \( l \) occurs in the instance)
- The weight vector \( \hat{w}_\ell \) 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 pivot feature $l$ in the source or the target

- Produce a correlation matrix $\mathbf{W}$

$$
\mathbf{W} = \begin{bmatrix} \hat{\mathbf{w}}_1 & \ldots & \hat{\mathbf{w}}_m \end{bmatrix}
$$
Computing Low Dim. Approximation

- SVD is employed to compute a low-dimensional linear approximation $\theta$

$$W = UDV^T \quad \theta = U^{T}[1:h,:)$$

- $\theta$: mapping from original space to new space

- The final set of features used for training and for testing: original features $x + \theta x$
Multi-task learning

- **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
One multi-task model: GO-MTL
(Kumar et al., ICML 2012)

- GO-MTL: Grouping and Overlap in Multi-Task Learning
- Does not assume that all tasks are related
- Applicable to classification and regression
GO-MTL assumptions

- All task models share **latent basic model components**

- Each task model is a **linear combination of shared latent components**

- The linear weight is **sparse**, to use few latent components
Notations

- $N$ tasks in total
- $k < N$ latent basis model components
- Each basis task is represented by a $I$ (a vector of size $d$)
- For all latent tasks, $L = (I_1, I_2, \ldots, I_k)$
- $L$ is learned from $N$ individual tasks.
  - E.g., weights/parameters of logistic regression or linear regression
The Approach

- $s^t$ is a linear weight vector and is assumed to be sparse.

\[
\theta^t = Ls^t
\]

- Stacking $s^t$ for all tasks, we get $S$. $S$ captures the task grouping structure.

\[
\theta_{d \times N} = L_{d \times k} \times S_{k \times N}
\]
Objective Function in GO-MTL

\[
\sum_{t=1}^{N} \sum_{i=1}^{n_t} \mathcal{L} \left( f(x_i^t; Ls^t), y_i^t \right) + \mu \| S \|_1 + \lambda \| L \|_F^2
\]
Optimization Strategy

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

$$s^t = \arg\min_{s} \sum_{i=1}^{n_t} \mathcal{L}(f(x_i^t; Ls), y_i^t) + \mu \|s\|_1$$

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

$$\arg\min_{L} \sum_{t=1}^{N} \sum_{i=1}^{n_t} \mathcal{L}(f(x_i^t; Ls^t), y_i^t) + \lambda \|L\|_F^2$$
Two tutorials on MTL

- Multi-Task Learning Primer. IJCNN’15, by Cong Li and Georgios C. Anagnostopoulos
Transfer, Multitask vs. Lifelong

- **Transfer learning vs. LML**
  - Transfer learning is not continuous
  - The source must be very similar to the target
  - No retention or accumulation of knowledge
  - Only one directional: help target domain

- **Multitask learning vs. LML**
  - Multitask learning retains no knowledge except data
  - Hard to re-learn all when tasks are numerous

- Incremental (online) multi-task learning is LML

KDD-2016
Online Learning

- The training data points come in a sequential order (online setting)
  - Computationally infeasible to train over the entire dataset
- Different from **LML**
  - Still performs the same learning task over time
  - LML aims to learn from a sequence of different tasks, retain and accumulate knowledge
Outline

- A motivating example
- What is lifelong machine learning?
- Related learning tasks
- **Lifelong supervised learning**
- Semi-supervised never-ending learning
- Lifelong unsupervised learning
- Lifelong reinforcement learning
- Summary
Lifelong Supervised Learning (LSL)

- The learner has performed learning on a sequence of supervised learning tasks, from 1 to $N$.

- When faced with the $(N+1)\text{th}$ task, it uses the relevant knowledge and labeled training data of the $(N+1)\text{th}$ task to help learning for the $(N+1)\text{th}$ task.
Early Work on Lifelong Learning
(Thrun, 1996b)

- **Concept learning tasks**: The functions are learned over the lifetime of the learner, $f_1, f_2, f_3, \ldots \in F$.

- Each task: learn the function $f: I \rightarrow \{0, 1\}$. $f(x) = 1$ means $x$ is a particular concept.
  - For example, $f_{\text{dog}}(x) = 1$ means $x$ is a dog.

- For $n$th task, we have its training data $X$
  - Also the training data $X_k$ of $k = 1, 2, \ldots, n-1$ tasks.
The paper proposed a few approaches based on two learning algorithms,

- Memory-based, e.g., kNN or Shepard’s method
- Neural networks,

**Intuition**: when we learn \( f_{\text{dog}}(x) \), we can use functions or knowledge learned from previous tasks, such as \( f_{\text{cat}}(x) \), \( f_{\text{bird}}(x) \), \( f_{\text{tree}}(x) \), etc.

- Data for \( f_{\text{cat}}(X) \), \( f_{\text{bird}}(X) \), \( f_{\text{tree}}(X) \)… 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 kNN.
  - Adjust \( g \) to minimize the energy function.

\[
E := \sum_{k=1}^{n-1} \left( \sum_{\langle x, y=1 \rangle \in X_k} \sum_{\langle x', y'=1 \rangle \in X_k} \|g(x) - g(x')\| - \sum_{\langle x', y'=0 \rangle \in X_k} \|g(x) - g(x')\| \right)
\]
  - \( g \) is a neural network, trained with Back-Prop.
  - kNN is then applied for the \( n \)th (new) task
Second Method

- It learns a distance function using support sets $d: I \times I \to [0, 1]$
  - It takes two input vectors $x$ and $x'$ from a pair of examples $<x, y>, <x', y'>$ of the same support set $X_k (k = 1, 2, \ldots, n-1)$
  - $d$ is trained with neural network using back-prop, and used as a general distance function
  - Training examples are:
    - $<(x, x'), 1>$ if $y = y' = 1$
    - $<(x, x'), 0>$ if $(y = 1 \land y' = 0)$ or $(y = 0 \land y' = 1)$
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, $<x_1, 1>, <x_2, 1>, \ldots \in X_n$ combined with Bayes’ rule

\[
P(f_n(x) = 1) = 1 - \left(1 + \prod_{(x', y'=1) \in X_n} \frac{d(x, x')}{1 - d(x, x')} \right)^{-1}
\]
LML Components in this case

- **KB**
  - **PIS**: store all the support sets.
  - **KS**: Distance function $d(x, x')$: the probability of example $x$ and $x'$ being the same concept.
    - Past knowledge is re-learned whenever a new task arrives.
  - **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)
Neural Network approaches
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 \( N \)th (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) \( N \)th task arrives, it first
  - selects the most similar cluster and then
  - uses the distance function of the cluster for classification in the \( N \)th task
Some Other Early works 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, incomplete 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 (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
  - Becomes 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).

\[
\sum_{t=1}^{T} \sum_{i=1}^{n_t} \mathcal{L} \left( f(x_i^{(t)}; Ls^{(t)}), y_i^{(t)} \right) + \mu \| S \|_1 + \lambda \| L \|_F^2
\]

- 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)

\[
e_T(L) = \frac{1}{T} \sum_{t=1}^{T} \min_{s^{(t)}} \left\{ \frac{1}{n_t} \sum_{i=1}^{n_t} \mathcal{L} \left( f \left( x_i^{(t)}; Ls^{(t)} \right), y_i^{(t)} \right) + \mu \| s^{(t)} \|_1 \right\} + \lambda \| L \|_F^2 , \quad (1)
\]
Approximate Equation (1)

- Eliminate the dependence on all of the past training data through inner summation
  - By using the second-order Taylor expansion of around $\theta = \theta^{(t)}$ where
  - $\theta^{(t)}$ is an optimal predictor learned on only the training data on task $t$. 
Taylor Expansion

- One variable function

\[ g(x) \approx g(a) + g'(a)(x - a) + \frac{1}{2} g''(a)(x - a)^2 \]

- Multivariate function

\[ g(x) \approx g(a) + \nabla g(a)(x - a) + \frac{1}{2} \| (x - a) \|^2 H(a) \]
Removing inner summation

\[ \frac{1}{N} \sum_{t=1}^{N} \min_{s_t} \left\{ \| \hat{\theta}^t - Ls_t \|_2^2 H_t + \mu \| s_t \|_1 \right\} + \lambda \| L \|_F^2 \]

\[ H^t = \frac{1}{2} \nabla^2_{\theta^t, \theta^t} \frac{1}{n_t} \sum_{i=1}^{n_t} \mathcal{L} (f(x^t_i; \theta^t), y^t_i) \mid_{\theta^t = \hat{\theta}^t} \]

\[ \hat{\theta}^t = \arg\min_{\theta^t} \frac{1}{n_t} \sum_{i=1}^{n_t} \mathcal{L} (f(x^t_i; \theta^t), y^t_i) \]
Simplify optimization

- **GO-MTL**: when computing a single candidate $L$, an optimization problem must be solved to recompute 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 $L$ will be changed.

- **Note**: (Ruvolo and Eaton, 2013b) added the mechanism to actively select the next task to learn.
ELLA Accuracy Result

- ELLA vs. GO-MTL

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Problem Type</th>
<th>Batch MTL Accuracy</th>
<th>ELLA Relative Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Land Mine</td>
<td>Classification</td>
<td>0.7802 ± 0.013 (AUC)</td>
<td>99.73 ± 0.7%</td>
</tr>
<tr>
<td>Facial Expr.</td>
<td>Classification</td>
<td>0.6577 ± 0.021 (AUC)</td>
<td>99.37 ± 3.1%</td>
</tr>
<tr>
<td>Syn. Data</td>
<td>Regression</td>
<td>−1.084 ± 0.006 (-rMSE)</td>
<td>97.74 ± 2.7%</td>
</tr>
<tr>
<td>London Sch.</td>
<td>Regression</td>
<td>−10.10 ± 0.066 (-rMSE)</td>
<td>98.90 ± 1.5%</td>
</tr>
</tbody>
</table>

*Batch MTL is GO-MTL*
### ELLA Speed Result

#### ELLA vs. GO-MTL

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Batch Runtime (seconds)</th>
<th>ELLA All Tasks (speedup)</th>
<th>ELLA New Task (speedup)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Land Mine</td>
<td>231±6.2</td>
<td>1,350±58</td>
<td>39,150±1,682</td>
</tr>
<tr>
<td>Facial Expr.</td>
<td>2,200±92</td>
<td>1,828±100</td>
<td>38,400±2,100</td>
</tr>
<tr>
<td>Syn. Data</td>
<td>1,300±141</td>
<td>5,026±685</td>
<td>502,600±68,500</td>
</tr>
<tr>
<td>London Sch.</td>
<td>715±36</td>
<td>2,721±225</td>
<td>378,219±31,275</td>
</tr>
</tbody>
</table>

**ELLA is 1K times faster than GO-MTL on all tasks, 30K times on a new task**
ELLA in LML

- **KB**
  - **PIS**: Stores all the task data
  - **KS**: matrix $L$ for $K$ basis tasks and $S$
    - Past knowledge is again re-learned whenever a new task arrives.
  - **KM**: optimization (e.g. alternating optimization strategy)

- **KBL**: Each task parameter vector is a linear combination of **KS**, i.e., $\theta^{(t)} = Ls^{(t)}$
Lifelong Sentiment Classification
(Chen, Ma, and Liu 2015)

“*I bought a cellphone 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)

- It 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 deal with domain dependent sentiment words and reliability of knowledge.
Naïve Bayesian Text Classification

- Key parameter

\[
P \left( w | c_j \right) = \frac{\lambda + N_{c_j,w}}{\lambda \cdot |V| + \sum_{v=1}^{\left| V \right|} N_{c_j,v}}
\]

- Only depends on the count of words in each class
LML Component: PIS

- Probabilities of a word appearing in positive or negative
  $$P^{\hat{t}}(w|+) \text{ and } P^{\hat{t}}(w|-)$$

- Word counts
  - Number of times that a word appears in positive class: $$N_{+,w}^{\hat{t}}$$
  - Number of times that a word appears in negative class: $$N_{-,w}^{\hat{t}}$$
LML Component: KB

- Two types of knowledge
  - Document-level knowledge
  - Domain-level knowledge
LML Component: KB

- Two types of knowledge
  - Document-level knowledge
  - Domain-level 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 $N^{KB}_{-,w} = \sum_{\hat{t}} N^{\hat{t}}_{-,w}$. 

LML Component: KB

- Two types of knowledge
  - Document-level knowledge
  - Domain-level knowledge

(b) Domain-level knowledge $M_{+,w}^{KB}$ (and $M_{-,w}^{KB}$): number of past tasks in which $P(w|+) > P(w|-)$ (and $P(w|+) < P(w|-)$).
LML Component: KM & KBL

- KM: performs counting and aggregation

- KBL: incorporates knowledge using regularization as penalty terms
Exploiting Knowledge via Penalties

- Penalty terms for two types of knowledge
  - Document-level knowledge
  - Domain-level knowledge
Exploiting Knowledge via Penalties

- Penalty terms for two types of knowledge
  - Document-level knowledge
  - Domain-level knowledge

\[
\frac{1}{2} \alpha \sum_{w \in V_T} \left( (X_{+,w} - N_{+,w}^t)^2 + (X_{-,w} - N_{-,w}^t)^2 \right)
\]

- \( t \) is the new task
Exploiting Knowledge via Penalties

- Penalty terms for two types of knowledge
  - Document-level knowledge
  - Domain-level knowledge

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

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

- $R_w$: ratio of #tasks where $w$ is positive / #all tasks
- $X^0_{+,w} = N^t_{+,w} + N^{KB}_{+,w}$ and $X^0_{-,w} = N^t_{-,w} + N^{KB}_{-,w}$
One Result of LSC model

- Better F1-score (left) and accuracy (right) with more past tasks
Cumulative Learning

- Cumulative learning (Fei et al., KDD-2016)
  - Open (World) Classification or Learning
    - Detecting unseen classes in testing
Toward self-learning

- Cumulative learning (Fei et al., KDD-2016)
  - Open (World) Classification or Learning
    - Detecting unseen classes in testing

- Incrementally adding new classes without re-training the whole model from scratch
  - At each time point, a new class is introduced.
  - The new task is the combination of all classes

- Self-learning: realizing something is new and learning it makes self-learning possible.
Based on space transformation

- Based on center-based similarity space (CBS) learning

- Each class has a center point and a circle range
  - Instances fall into it are more likely to belong to this class.
Main steps

- Search for a set of classes SC that are similar to the new \((N + 1)\) class

- Learn to separate the new class and the classes in SC

- Build a new model for the new class, update the models for classes in SC
Outline

- A motivating example
- What is lifelong machine learning?
- Related learning tasks
- Lifelong supervised learning
- Semi-supervised never-ending learning
- Lifelong unsupervised learning
- Lifelong reinforcement learning
- Summary
Humans learn many things, for years, and become better learners over time

Why not machines?
Never-Ending Learning

We’ll never really understand learning until we build machines that

• learn many different things,

• over years,

• and become better learners over time.
Never-Ending Learning

We’ll never produce natural language understanding systems until we have systems that react to arbitrary sentences by saying one of:

- I understand, and already knew that
- I understand, and didn’t know, but accept it
- I understand, and disagree because …
Main Task: acquire a growing competence without asymptote

- over years
- multiple functions
- where learning one thing improves ability to learn the next
- acquiring data from humans, environment

Many candidate domains:

- Robots
- Softbots
- Game players
NELL: Never-Ending Language Learner

Inputs:
- initial ontology
- handful of examples of each predicate in ontology
- the web
- occasional interaction with human trainers

The task:
- run 24x7, forever
  - each day:
    1. extract more facts from the web to populate the initial ontology
    2. learn to read (perform #1) better than yesterday
NELL: Never-Ending Language Learner

Goal:
• run 24x7, forever
• each day:
  1. extract more facts from the web to populate given ontology
  2. learn to read better than yesterday

Today...
Running 24 x 7, since January, 2010

Input:
• ontology defining ~800 categories and relations
• 10-20 seed examples of each
• 1 billion web pages (ClueWeb – Jamie Callan)

Result:
• continuously growing KB with +90,000,000 extracted beliefs
Read the Web
Research Project at Carnegie Mellon University

NELL: Never-Ending Language Learning

Can computers learn to read? We think so. "Read the Web" is a research project that attempts to create a computer system that learns over time to read the web. Since January 2010, our computer system called NELL (Never-Ending Language Learner) has been running continuously, attempting to perform two tasks each day:

- First, it attempts to "read," or extract facts from text found in hundreds of millions of web pages (e.g., `playsInstrument(George_Harrison, guitar)`).
- Second, it attempts to improve its reading competence, so that tomorrow it can extract more facts from the web, more accurately.

So far, NELL has accumulated over 15 million candidate beliefs by reading the web, and it is considering these at different levels of confidence. NELL has high confidence in 1,471,011 of these beliefs — these are displayed on this website. It is not perfect, but NELL is learning. You can track NELL's progress below or @cmunell on Twitter, browse and download its knowledge base, read more about our technical approach, or join the discussion group.
## Recently-Learned Facts

<table>
<thead>
<tr>
<th>Instance</th>
<th>Iteration</th>
<th>Date Learned</th>
<th>Confidence</th>
</tr>
</thead>
<tbody>
<tr>
<td>bob_ford is a journalist</td>
<td>941</td>
<td>25-jul-2015</td>
<td>100.0</td>
</tr>
<tr>
<td>wgc_hsbc_champions is an award, championship, or tournament trophy</td>
<td>941</td>
<td>25-jul-2015</td>
<td>97.0</td>
</tr>
<tr>
<td>elizabeth_cotten is a European person</td>
<td>941</td>
<td>25-jul-2015</td>
<td>99.8</td>
</tr>
<tr>
<td>n1_17 is a dataset used within the scientific field of machine learning</td>
<td>941</td>
<td>25-jul-2015</td>
<td>100.0</td>
</tr>
<tr>
<td>mycorrhizal_fungi is a bacterium</td>
<td>941</td>
<td>25-jul-2015</td>
<td>100.0</td>
</tr>
<tr>
<td>eric_byrnes is an athlete who led utah jazz jerseys</td>
<td>946</td>
<td>03-sep-2015</td>
<td>99.6</td>
</tr>
<tr>
<td>Cabrillo High School Aquarium is an aquarium in the city Lompoc</td>
<td>946</td>
<td>03-sep-2015</td>
<td>100.0</td>
</tr>
<tr>
<td>state university is a sports team also known as Michigan State University</td>
<td>944</td>
<td>11-aug-2015</td>
<td>100.0</td>
</tr>
<tr>
<td>molluscs is called clams</td>
<td>944</td>
<td>11-aug-2015</td>
<td>99.1</td>
</tr>
<tr>
<td>pulmonary artery arises from aorta</td>
<td>946</td>
<td>03-sep-2015</td>
<td>100.0</td>
</tr>
</tbody>
</table>
Computer Reading the Web

1. Classify noun phrases (NP’s) by category
The Problem with Semi-Supervised Bootstrap Learning

- Paris
- Pittsburgh
- Seattle
- Cupertino
The Problem with Semi-Supervised Bootstrap Learning

- Paris
- Pittsburgh
- Seattle
- Cupertino

- Humans never learn in isolation
- We learn effectively from a few examples with the help of the past knowledge.
The Problem with Semi-Supervised Bootstrap Learning

- Paris
- Pittsburgh
- Seattle
- Cupertino

- mayor of arg1
- live in arg1
The Problem with Semi-Supervised Bootstrap Learning

- Paris
- Pittsburgh
- Seattle
- Cupertino
- San Francisco
- Austin
- mayor of arg1
- live in arg1
- denial
The Problem with Semi-Supervised Bootstrap Learning

- Paris
- Pittsburgh
- Seattle
- Cupertino
- mayor of arg1
- live in arg1

- San Francisco
- Austin
- denial

- arg1 is home of
- traits such as arg1
The Problem with Semi-Supervised Bootstrap Learning

- Paris
- Pittsburgh
- Seattle
- Cupertino
- San Francisco
- Austin
- London

- mayor of arg1
- live in arg1
- anxiety
- selfishness
- denial
- arg1 is home of
- traits such as arg1

KDD-2016
The Problem with Semi-Supervised Bootstrap Learning

Paris
Pittsburgh
Seattle
Cupertino

San Francisco
Austin

London

mayor of arg1
live in arg1
arg1 is home of

anxiety
selfishness

denial

traits such as arg1
Key Idea 1: Coupled semi-supervised training of many functions

hard (underconstrained) semi-supervised learning problem

much easier (more constrained) semi-supervised learning problem
Key Idea 1: Coupled semi-supervised training of many functions

Let’s call this: Machine Learning (ML) 1.0

- Isolated learning has limitations.
Key Idea 1: Coupled semi-supervised training of many functions

Let’s call this: Machine Learning (ML) 1.0

- Isolated learning has limitations.

- It is rather “silly” not to exploit such sharing in learning or extraction.
Coupled Training Type 1: Co-training, Multiview, Co-regularization

[Blum & Mitchell; 98]
[Dasgupta et al; 01 ]
[Ganchev et al., 08]
[Sridharan & Kakade, 08]
[Wang & Zhou, ICML10]
Coupled Training Type 1: Co-training, Multiview, Co-regularization

Constraint: \( f_1(x_1) = f_2(x_2) \)
Coupled Training Type 1: Co-training, Multiview, Co-regularization

If $f_1, f_2$ PAC learnable, $X_1, X_2$ conditionally indep
Then PAC learnable from unlabeled data and weak initial learner

Constraint: $f_1(x_1) = f_2(x_2)$ and disagreement between $f_1, f_2$ bounds error of each

[Blum & Mitchell; 98]
[Dasgupta et al; 01]
[Ganchev et al., 08]
[Sridharan & Kakade, 08]
[Wang & Zhou, ICML10]
Type 1 Coupling Constraints in NELL

NP:
- NP text context distribution
- NP morphology
- NP HTML contexts

f₁(NP) → person
f₂(NP)
f₃(NP)

www.celebrities.com:
  <li>__</li>
Coupled Training Type 2:
Structured Outputs, Multitask, Posterior Regularization, Multilabel

\[
\Phi(Y_1, Y_2)
\]

Constraint: \( \Phi(f_1(x), f_2(x)) \)

References:
[Daume, 2008]
[Bakhir et al., eds. 2007]
[Roth et al., 2008]
[Taskar et al., 2009]
[Carlson et al., 2009]
Coupled Training Type 2: Structured Outputs, Multitask, Posterior Regularization, Multilabel

\[ \Phi(Y_1, Y_2) \]

\[ f_1(x) \]

\[ f_2(x) \]

\[ X \]

\[ Y_1 \]

\[ Y_2 \]

Effectiveness \sim probability that \( \Phi(Y_1, Y_2) \) will be violated by incorrect \( f_j \) and \( f_k \)

Constraint: \( \Phi(f_1(x), f_2(x)) \)

References:
- Daume, 2008
- Bakhir et al., eds. 2007
- Roth et al., 2008
- Taskar et al., 2009
- Carlson et al., 2009
Type 2 Coupling Constraints in NELL

- **Green constraints:**
  - \( \text{athlete} \rightarrow \text{person} \)

- **Red constraints:**
  - \( \text{athlete} \rightarrow \text{NOT} \ \text{sport} \)
  - \( \text{NOT} \ \text{athlete} \leftarrow \text{sport} \)
Multi-view, Multi-Task Coupling

NP:

- NP text context distribution
- NP morphology
- NP HTML contexts
Computer Reading the Web

1. Classify noun phrases (NP’s) by category
2. Classify NP pairs by relation
Learning Relations between NP’s

NP1

playsSport(a,s)

playsForTeam(a,t)

teamPlaysSport(t,s)

coachesTeam(c,t)

NP2
Learning Relations between NP’s
Type 3 Coupling: Argument Types

Constraint: \( f_3(x_1, x_2) \Rightarrow (f_1(x_1) \text{ AND } f_2(x_2)) \)

playsSport(a, s)
playsForTeam(a, t)

coachesTeam(c, t)
teamPlaysSport(t, s)

athlete
person
sport

NP1
NP2

playsSport(NP1, NP2) \Rightarrow \text{athlete}(NP1), \text{sport}(NP2)
Pure EM Approach to Coupled Training

$E$: jointly estimate latent labels for each function of each unlabeled example

$M$: retrain all functions, based on these probabilistic labels

Scaling problem:
- $E$ step: 20M NP’s, $10^{14}$ NP pairs to label
- $M$ step: 50M text contexts to consider for each function $10^{10}$ parameters to retrain
- even more URL-HTML contexts..
NELL’s Approximation to EM

E’ step:
- Consider only a growing subset of the latent variable assignments
  - category variables: up to 250 NP’s per category per iteration
  - relation variables: add only if confident and args of correct type
  - this set of explicit latent assignments *IS* the knowledge base

M’ step:
- Each view-based learner retrains itself from the updated KB
- “context” methods create growing subsets of contexts
Never-Ending Language Learning

arg1_was_playing_arg2_arg2_megastar_arg1_arg2_icons_arg1
arg2_player_named_arg1_arg2_prodigy_arg1
arg1_is_the_tiger_woods_of_arg2_arg2_career_of_arg1
arg2_greats_as_arg1_arg1_plays_arg2_arg2_player_is_arg1
arg2_legends_arg1_arg1_announced_his_retirement_from_arg2
arg2_operations_chief_arg1_arg2_player_like_arg1
arg2_and_golfing_personalities_including_arg1_arg2_players_like_arg1
arg2_greats_like_arg1_arg2_players_are_steffi_graf_and_arg1
arg2_great_arg1_arg2_champ_arg1_arg2_greats_such_as_arg1
arg2_professionals_such_as_arg1_arg2_hit_by_arg1_arg2_greats_arg1
arg2_icon_arg1_arg2_stars_like_arg1_arg2_pros_like_arg1
arg1_retires_from_arg2_arg2_phenom_arg1_arg2_lesson_from_arg1
arg2_architects_robert_trent_jones_and_arg1_arg2_sensation_arg1
arg2_pros_arg1_arg1_stars_venus_and_arg1_arg2_hall_of_famer_arg1
arg2_superstar_arg1_arg2_legend_arg1_arg2_legends_such_as_arg1
arg2_players_is_arg1_arg2_pro_arg1_arg2_player_was_arg1
arg2_god_arg1_arg2_idol_arg1_arg1_was_born_to_play_arg2
arg2_star_arg1_arg2_hero_arg1_arg2_players_are_arg1
arg1_retired_from_professional_arg2_arg2_legends_as_arg1
arg2_autographed_by_arg1_arg2_champion_arg1

<table>
<thead>
<tr>
<th>Predicate</th>
<th>Feature</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>mountain</td>
<td>LAST=peak</td>
<td>1.791</td>
</tr>
<tr>
<td>mountain</td>
<td>LAST=mountain</td>
<td>1.093</td>
</tr>
<tr>
<td>mountain</td>
<td>FIRST=mountain</td>
<td>-0.875</td>
</tr>
<tr>
<td>musicArtist</td>
<td>POS=DT_NNS</td>
<td>0.141</td>
</tr>
<tr>
<td>musicArtist</td>
<td>POS=DT_JJ_NN</td>
<td>-0.807</td>
</tr>
<tr>
<td>newspaper</td>
<td>LAST=sun</td>
<td>1.330</td>
</tr>
<tr>
<td>newspaper</td>
<td>LAST=university</td>
<td>-0.318</td>
</tr>
<tr>
<td>university</td>
<td>POS=NN_NNS</td>
<td>-0.798</td>
</tr>
<tr>
<td>university</td>
<td>LAST=college</td>
<td>2.076</td>
</tr>
<tr>
<td>university</td>
<td>PREFIX=uc</td>
<td>1.999</td>
</tr>
<tr>
<td>university</td>
<td>LAST=state</td>
<td>1.992</td>
</tr>
<tr>
<td>university</td>
<td>LAST=university</td>
<td>1.745</td>
</tr>
<tr>
<td>university</td>
<td>FIRST=college</td>
<td>-1.381</td>
</tr>
<tr>
<td>visualArtMovement</td>
<td>SUFFIX=ism</td>
<td>1.282</td>
</tr>
<tr>
<td>visualArtMovement</td>
<td>PREFIX=journ</td>
<td>-0.234</td>
</tr>
<tr>
<td>visualArtMovement</td>
<td>PREFIX=budd</td>
<td>-0.253</td>
</tr>
</tbody>
</table>

Predicate | Web URL                                      | Extraction Template                                      |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>academicField</td>
<td><a href="http://scholendow.ais.msu.edu/student/ScholSearch.Asp">http://scholendow.ais.msu.edu/student/ScholSearch.Asp</a></td>
<td> [X] - &lt;a href='d.author.aspx?a=[X]'&gt;&lt;option&gt;[X]&lt;/option&gt;</td>
</tr>
</tbody>
</table>
Never-Ending Language Learning

Humans never learn in isolation

We learn effectively from a few examples with the help of the past knowledge.

Predicate Feature Weight

mountain LASTname 1.791
1.093
-0.875
1.853
1.412
-0.807
1.330
-0.318
-0.798
2.076

university PREFIX=uc 1.999
university LAST=state 1.992
university LAST=university 1.745
university FIRST=college -1.381
visualArtMovement SUFFIX=ism 1.282
visualArtMovement PREFIX=journ -0.234
visualArtMovement PREFIX=budd -0.253

Predicate Web URL
academicField http://scholendow.ais.msu.edu/student/ScholSearch.Asp
bird http://www.michaelforsberg.com/stock.html
bookAuthor http://lifebehindthecurve.com/
1. Classify noun phrases (NP’s) by category
2. Classify NP pairs by relation
3. Discover rules to predict new relation instances
Key Idea 2: Discover New Coupling Constraints

• first order, probabilistic horn clause constraints

\[
0.93 \text{ athletePlaysSport}(\ ?x, \ ?y) : - \text{athletePlaysForTeam}(\ ?x, \ ?z), \\
\text{teamPlaysSport}(\ ?z, \ ?y)
\]

– connects previously uncoupled relation predicates
– infers new beliefs for KB
Example Learned Horn Clauses

0.95 athletePlaysSport(?x,basketball) :- athleteInLeague(?x,NBA)

0.93 athletePlaysSport(?x,?y) :- athletePlaysForTeam(?x,?z)
   teamPlaysSport(?z,?y)

0.91 teamPlaysInLeague(?x,NHL) :- teamWonTrophy(?x,Stanley_Cup)

0.90 athleteInLeague(?x,?y):-athletePlaysForTeam(?x,?z),
   teamPlaysInLeague(?z,?y)

0.88 cityInState(?x,?y) :- cityCapitalOfState(?x,?y),
   cityInCountry(?y,USA)

0.62* newspaperInCity(?x,New_York) :- companyEconomicSector(?x,media),
   generalizations(?x,blog)
Learned Probabilistic Horn Clause Rules
Learned Probabilistic Horn Clause Rules

0.93 \( \text{playsSport}(\text{x}, \text{y}) \leftarrow \text{playsForTeam}(\text{x}, \text{z}), \text{team PlaysSport}(\text{z}, \text{y}) \)
1. Classify noun phrases (NP’s) by category
2. Classify NP pairs by relation
3. Discover rules to predict new relation instances
4. Learn which NP’s (co)refer to which latent concepts
Distinguish Text Tokens from Entities

Text Tokens

- Apple_theNP
- AppleInc_theNP

Entities

- Apple_theFruit
- Apple_theCompany

Coreference Resolution:

- Co-train classifier to predict coreference as $f$(string similarity, extracted beliefs)
- Small amount of supervision: ~10 labeled coreference decisions
- Cluster tokens using $f$ as similarity measure
Computer Reading the Web

1. Classify noun phrases (NP’s) by category
2. Classify NP pairs by relation
3. Discover rules to predict new relation instances
4. Learn which NP’s (co)refer to which latent concepts
5. Discover new relations to extend ontology
OntExt (Ontology Extension)
OntExt (Ontology Extension)
OntExt (Ontology Extension)

- Everything
  - Company
    - WorksFor
  - Person
    - Plays
  - Sport
    - PlayedIn
  - City
    - LocatedIn
Prophet

- Mining the Graph representing NELL’s KB to:
  1. Extend the KB by predicting new relations (edges) that might exist between pairs of nodes;
  2. Induce inference rules;
  3. Identify misplaced edges which can be used by NELL as hints to identify wrong connections between nodes (wrong fats);
Prophet

- Find open triangles in the Graph

Appel and Hruschka, 2011
Prophet

• Find open triangles in the Graph
Prophet

- open triangles

Diagram:

- Milwaukee Bucks
  - sportTeam
  - teamPlaysInLeague

- Basketball
  - Sport

- NBA
  - Sport’s League
open triangles
Prophet

- open triangles

Diagram:

- Pittsburgh Penguins
  - sportTeam: Hokey
  - teamPlaysInLeague: NHL
open triangles
Prophet

- open triangles

Diagram:
- Dallas Cowboys
  - sportTeam: Football
  - teamPlaysInLeague: NFL
open triangles
Prophet

- open triangles
Prophet

- open triangles
Prophet

- open triangles

- Name the new relation based on a big textual corpus
<table>
<thead>
<tr>
<th>Contexts/Contexts</th>
<th>may cause</th>
<th>can cause</th>
<th>can lead to</th>
<th>to treat</th>
<th>for treatment of</th>
<th>medication</th>
</tr>
</thead>
<tbody>
<tr>
<td>may cause</td>
<td>0.176</td>
<td>0.074</td>
<td>0.030</td>
<td>0.015</td>
<td>0.011</td>
<td>0.000</td>
</tr>
<tr>
<td>can cause</td>
<td>0.051</td>
<td>0.150</td>
<td>0.039</td>
<td>0.018</td>
<td>0.013</td>
<td>0.010</td>
</tr>
<tr>
<td>can lead to</td>
<td>0.034</td>
<td>0.064</td>
<td>0.189</td>
<td>0.019</td>
<td>0.021</td>
<td>0.018</td>
</tr>
<tr>
<td>to treat</td>
<td>0.006</td>
<td>0.011</td>
<td>0.007</td>
<td>0.109</td>
<td>0.043</td>
<td>0.015</td>
</tr>
<tr>
<td>for treatment of</td>
<td>0.005</td>
<td>0.008</td>
<td>0.008</td>
<td>0.045</td>
<td>0.086</td>
<td>0.023</td>
</tr>
<tr>
<td>medication</td>
<td>0.000</td>
<td>0.011</td>
<td>0.009</td>
<td>0.030</td>
<td>0.036</td>
<td>0.111</td>
</tr>
</tbody>
</table>

Clustering:

- (Vioxx, Arthritis)
- (Fosamax, Osteoporosis)
- (Metformin, diabetes)
- (Singulair, Asthma)

- 'to treat'
- 'for treatment of'
- 'medication'

- 'can cause'
- 'may cause'
- 'leads to'

- (Marijuana, Cancer)
- (Prozac, Migranes)
- (Paxil, Diarrhea)
NELL: sample of self-added relations

- athleteWonAward
- animalEatsFood
- languageTaughtInCity
- clothingMadeFromPlant
- beverageServedWithFood
- fishServedWithFood
- athleteBeatAthlete
- athleteInjuredBodyPart
- arthropodFeedsOnInsect
- animalEatsVegetable
- plantRepresentsEmotion
- foodDecreasesRiskOfDisease
- clothingGoesWithClothing
- bacteriaCausesPhysCondition
- buildingMadeOfMaterial
- emotionAssociatedWithDisease
- foodCanCauseDisease
- agriculturalProductAttractsInsect
- arteryArisesFromArtery
- countryHasSportsFans
- bakedGoodServedWithBeverage
- beverageContainsProtein
- animalCanDevelopDisease
- beverageMadeFromBeverage
1. Classify noun phrases (NP’s) by category
2. Classify NP pairs by relation
3. Discover rules to predict new relation instances
4. Learn which NP’s (co)refer to which latent concepts
5. Discover new relations to extend ontology
6. Learn to infer relation instances via targeted random walks (PRA)
CityLocatedInCountry(Pittsburgh) = ?

- Pittsburgh

- **Feature** = Typed Path
  - CityInState, CityInstate\(^{-1}\), CityLocatedInCountry

- **Feature Value**

- **Logistic Regression Weight**
  - 0.32

[Lao, Mitchell, Cohen, *EMNLP 2011*]
CityLocatedInCountry(Pittsburgh) = ?

- Feature = Typed Path
  - CityInState, CityInState⁻¹, CityLocatedInCountry

- Feature Value
  - Logistic Regression Weight

- Logistic Regression Weight
  - 0.32
CityLocatedInCountry(Pittsburgh) = ?

- Feature = Typed Path
  - CityInState, CityInState\(^{-1}\), CityLocatedInCountry

- Feature Value
  - Weight: 0.32

[Feature Value]

- Logistic Regression

[Lao, Mitchell, Cohen, *EMNLP 2011*]
Feature = Typed Path
CityInState, CityInstate⁻¹, CityLocatedInCountry

Feature Value
CityLocatedInCountry(Pittsburgh) = ?

Logistic Regression Weight
0.32

[La, Mitchell, Cohen, EMNLP 2011]
Feature = Typed Path

CityInState, CityInState⁻¹, CityLocatedInCountry

Feature Value

Pr(U.S. | Pittsburgh, TypedPath)

Weight

Logistic Regression

Pr(U.S. | Pittsburgh, TypedPath) = ?

[Feature = Typed Path]

CityLocatedInCountry(Pittsburgh) = ?

[Feature Value]

Pennsylvania

CityInState

Philadelphia

CityLocatedInCountry

U.S.

CityInState⁻¹

CityInState

Harisburg

Pittsburgh

...(14)

[Logistic Regression Weight] 0.32

[Feature Value] 0.8

[Lao, Mitchell, Cohen, EMNLP 2011]
CityLocatedInCountry(Pittsburgh) = ?

Feature = Typed Path

- CityInState, CityInstate⁻¹, CityLocatedInCountry
- AtLocation⁻¹, AtLocation, CityLocatedInCountry

Feature Value

- CityLocatedInCountry(Pittsburgh) = 0.8

Logistic Regression Weight

- 0.32
- 0.20

[Laos, Mitchell, Cohen, EMNLP 2011]
CityLocatedInCountry(Pittsburgh) = ?

Feature = Typed Path
CityInState, CityInState⁻¹, CityLocatedInCountry
AtLocation⁻¹, AtLocation, CityLocatedInCountry

Feature Value
CityLocatedInCountry(Pittsburgh) = ?

[Logistic Regression]

Feature Value
CityLocatedInCountry
AtLocation
PPG
Delta

Weight
0.32
0.20

[Source: Lao, Mitchell, Cohen, EMNLP 2011]
CityLocatedInCountry(Pittsburgh) = ?

Feature = Typed Path
- CityInState, CityInstate⁻¹, CityLocatedInCountry
- AtLocation⁻¹, AtLocation, CityLocatedInCountry

Feature Value
- Logisitic Regression Weight
  - CityLocatedInCountry(Pittsburgh) = ?
  - 0.8 0.32
  - Pittsburgh
  - Philadelphia
  - Harisburg
  -... (14)
  - U.S.
  - Pittsburgh
  - Philadelphia
  - Harisburg
  -... (14)
  - AtLocation⁻¹
  - AtLocation
  - PPG
  - Delta
  - Atlanta
  - Dallas
  - Tokyo

[Laas, Mitchell, Cohen, EMNLP 2011]
CityLocatedInCountry(Pittsburgh) = ?

- **Feature = Typed Path**
  - CityInState, CityInstate\(^{-1}\), CityLocatedInCountry
  - AtLocation\(^{-1}\), AtLocation, CityLocatedInCountry

- **Feature Value**
  - 0.8
  - 0.6

- **Logistic Regression Weight**
  - 0.32
  - 0.20

[Lao, Mitchell, Cohen, EMNLP 2011]
Feature = Typed Path
- CityInState, CityInState⁻¹, CityLocatedInCountry
- AtLocation⁻¹, AtLocation, CityLocatedInCountry
- ...

Feature value
- CityLocatedInCountry(Pittsburgh) = U.S. p=0.58

1. Tractable (bounded length)
2. Anytime
3. Accuracy increases as KB grows
4. Combines probabilities from different horn clauses

Logistic Regression
- Feature = Typed Path
- CityInState, CityInState⁻¹, CityLocatedInCountry
- AtLocation⁻¹, AtLocation, CityLocatedInCountry
- ...
Random walk inference: learned rules

CityLocatedInCountry(*city, country*):

8.04 cityliesonriver, cityliesonriver\(^{-1}\), citylocatedincountry
5.42 hasofficeincity\(^{-1}\), hasofficeincity, citylocatedincountry
4.98 cityalsoknownas, cityalsoknownas, citylocatedincountry
2.85 citycapitalofcountry, citylocatedincountry\(^{-1}\), citylocatedincountry
2.29 agentactsinlocation\(^{-1}\), agentactsinlocation, citylocatedincountry
1.22 statehascapital\(^{-1}\), statelocatedincountry
0.66 citycapitalofcountry

7 of the 2985 learned rules for CityLocatedInCountry
Key Idea 4: Cumulative, Staged Learning
Learning X improves ability to learn Y

1. Classify noun phrases (NP’s) by category
2. Classify NP pairs by relation
3. Discover rules to predict new relation instances
4. Learn which NP’s (co)refer to which latent concepts
5. Discover new relations to extend ontology
6. Learn to infer relation instances via targeted random walks (PRA)
7. Vision: connect NELL and NEIL
8. Mutilingual NELL (Portuguese)
9. CrossLingual NELL
10. Learn to microread single sentences
11. Self reflection, self-directed learning
12. Goal-driven reading: predict, then read to corroborate/correct
13. Make NELL learn by conversation (e.g, Twitter)
14. Add a robot body, or mobile phone body, to NELL
Key Idea 4: Cumulative, Staged Learning
Learning X improves ability to learn Y

1. Classify noun phrases (NP’s) by category
2. Classify NP pairs by relation
3. Discover rules to predict new relation instances
4. Learn which NP’s (co)refer to which latent concepts
5. Discover new relations to extend ontology
6. Learn to infer relation instances via targeted random walks
7. Vision: connect NELL and NEIL
8. Multilingual NELL (Portuguese)

9. CrossLingual NELL
10. Learn to microread single sentences
11. Self reflection, self-directed learning
12. Goal-driven reading: predict, then read to corroborate/correct
13. Make NELL learn by conversation (e.g., Twitter)
14. Add a robot body, or mobile phone body, to NELL
NELL Architecture

Knowledge Base (latent variables)

Beliefs

Candidate Beliefs

Knowledge Integrator

Text Context patterns (CPL)

Orthographic classifier (CML)

URL specific HTML patterns (SEAL)

Human advice

Actively search for web text (OpenEval)

Infer new beliefs from old (PRA)

Image classifier (NEIL)

Ontology extender (OntExt)

KDD-2016
Conversing Learning
Conversing Learning

- Help to supervise NELL by automatically asking questions on Web Communities
Conversing Learning

- Help to supervise NELL by automatically asking questions on Web Communities
Conversing Learning

- Uses an agent (SS-Crowd) capable of:
  - building questions;
  - Posting questions in Web communities;
  - Fetch answers;
  - Understand the answers;
  - Decide on how much to believe on the answers
Conversing Learning

Pedro & Hruschka
Question: (Yes or No?) If athlete Z is member of team X and athlete Z plays in league Y, then team X plays in league Y.

Twitter answers sample:

No. \((Z \text{ in } X) \land (Z \text{ in } Y) \rightarrow (X \text{ in } Y)\)

Yahoo! Answers sample:

NO, Not in EVERY case. Athlete Z could be a member of football team X and he could also play in his pub’s Friday nights dart team. The Dart team could play in league Y (and Z therefore by definition plays in league Y). This does not mean that the football team plays in the darts league!
In the word sequence "Pittsburgh Steelers beat X", could X be a sports team?

A) it could only be a sports team
B) it could be a sports team or something else
C) it's probably not a sports team
D) the sequence does not make sense
Lifelong Learning components

- **Past information store (PIS):** It stores previously extracted results, phrasings, morphological features, and web page structures.
- **Knowledge reasoner (KR):** Path Ranking Algorithm PRA.
- **Knowledge-based learner (KBL):** Semi-supervised learning using initial and new information in PIS with the help of coupling constraints. It also has a knowledge integrator.
Lifelong Learning components

- Past information store (PIS): It stores previously extracted results, phrasings, morphological features, and web page structures.
- Knowledge reasoner (KR): A first-order relational learning system.
- Knowledge-based learner (KBL): Semi-supervised learning using initial and new information in PIS with the help of coupling constraints. It also has a knowledge integrator.

Key Characteristics of LML

- Continuous learning process
- Knowledge accumulation in KB
- Use of past knowledge to help future learning
15 Minutes Break
Outline

- A motivating example
- What is lifelong machine learning?
- Related learning tasks
- Lifelong supervised learning
- Semi-supervised never-ending learning
- Lifelong unsupervised learning
- Lifelong reinforcement learning
- Summary
**LTM: Lifelong Topic Modeling**  
(Chen and Liu, ICML-2014)

- Topic modeling (Blei et al 2003) finds 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, \ldots\}
LTM: Lifelong Topic Modeling
(Chen and Liu, ICML-2014)

- Topic modeling (Blei et al 2003) finds 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
“The size is great, but pictures are poor.”

- Aspects (product features): size, picture

Why lifelong learning can help SA?

- 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 setting

- Given a large set of document collections (big data), \( D = \{D_1, D_2, \ldots, D_N\} \), learn from each \( D_i \) to produce the results \( S_i \). Let \( S = \bigcup_i S_i \).
  - \( S \) is called 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 \( D^t \) not in \( D \)
  - The results learned this way should be better than those without the guidance of \( S \) (and \( D \))
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**: Run a topic model (e.g., LDA) on each domain $D_i$ to produce a set of topics $S_i$ called **Topic Base**

■ **Step 2**: Mine prior knowledge (**must-links**) and use knowledge to guide modeling.
Algorithm 2 LTM($D^t$, $S$)

1: $A^t \leftarrow \text{GibbsSampling}(D^t, \emptyset, N)$; // Run $N$ Gibbs iterations with no knowledge (equivalent to LDA).
2: \textbf{for} $i = 1$ \textbf{to} $N$ \textbf{do}
3: \hspace{1em} $K^t \leftarrow \text{KnowledgeMining}(A^t, S)$;
4: \hspace{1em} $A^t \leftarrow \text{GibbsSampling}(D^t, K^t, 1)$; // Run with knowledge $K^t$.
5: \textbf{end for}
Knowledge Mining Function

- **Topic matching**: find similar topics from topic base for each topic in the new domain

- **Pattern mining**: find frequent itemsets from the matched topics
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, expensive\}
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, expensive\}

- If we require words to 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.
Algorithm 3 KnowledgeMining($A^t, S$)

1: for each p-topic $s_k \in S$ do
2: \hspace{1em} $j^* = \min_j \text{KL-Divergence}(a_j, s_k)$ for $a_j \in A^t$;
3: \hspace{1em} if KL-Divergence($a_{j^*}, s_k$) $\leq \pi$ then
4: \hspace{2em} $M^t_{j^*} \leftarrow M^t_{j^*} \cup s_k$;
5: \hspace{1em} end if
6: end for
7: $K^t \leftarrow \cup_{j^*} \text{FIM}(M^t_{j^*})$; // Frequent Itemset Mining.
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)
Figure 2. Top & Middle: Topical words Precision@5 & Precision@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
LML components of LTM

- Knowledge Base (KB)
  - Past information store (PIS): It stores topics/aspects generated in the past tasks
    - Also called topic base
  - Knowledge store (KS): It contains knowledge mined from PIS: **Must-Links**
  - Knowledge miner (KM): Frequent pattern mining using past topics as transactions

- Knowledge-based learner (KBL): LTM is based on **Generalized Pólya Urn Model**
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 PIS without considering the target task/data
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
Cannot-links are mined in each Gibbs iteration
Overall Algorithm

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

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

- Sampling becomes much more complex
  - It proposed M-GPU model (multi-generalized Polya urn model)
### AMC results

<table>
<thead>
<tr>
<th>Price</th>
<th>Size &amp; Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AMC</strong></td>
<td><strong>LTM</strong></td>
</tr>
<tr>
<td>money</td>
<td>shot</td>
</tr>
<tr>
<td>buy</td>
<td>money</td>
</tr>
<tr>
<td>price</td>
<td>review</td>
</tr>
<tr>
<td>range</td>
<td>price</td>
</tr>
<tr>
<td>cheap</td>
<td>cheap</td>
</tr>
<tr>
<td>expensive</td>
<td>camcorder</td>
</tr>
<tr>
<td>deal</td>
<td>condition</td>
</tr>
<tr>
<td>point</td>
<td>con</td>
</tr>
<tr>
<td>performance</td>
<td>sony</td>
</tr>
<tr>
<td>extra</td>
<td>trip</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>AMC</td>
<td>LTM</td>
</tr>
<tr>
<td>size</td>
<td>small</td>
</tr>
<tr>
<td>small</td>
<td>big</td>
</tr>
<tr>
<td>smaller</td>
<td>size</td>
</tr>
<tr>
<td>weight</td>
<td>pocket</td>
</tr>
<tr>
<td>compact</td>
<td>hand</td>
</tr>
<tr>
<td>hand</td>
<td>big</td>
</tr>
<tr>
<td>big</td>
<td>pocket</td>
</tr>
<tr>
<td>pocket</td>
<td>screen</td>
</tr>
<tr>
<td>heavy</td>
<td>case</td>
</tr>
<tr>
<td>case</td>
<td>exposure</td>
</tr>
</tbody>
</table>

**Table 2:** Example topics of AMC, LTM and LDA from the Camera domain. Errors are italicized and marked in red.
Lifelong Learning components

- Knowledge Base (KB)
  - Past information store (PIS): It stores topics/aspects generated in the past tasks
  - Knowledge store (KS): It contains knowledge mined from PIS: must-links and cannot-links
  - Knowledge miner (KM): Frequent pattern mining & …

- Knowledge-based learner (KBL): LTM based on multi-generalized Polya urn Model
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
  - Makes *lifelong learning* promising

- **Data:** a huge volume of reviews of all kinds
LAST Model

- Lifelong aspect-based sentiment topic model (Wang et al., 2016)

Knowledge

- Aspect-opinion pair, e.g., {shipping, quick}
- Aspect-aspect pair, e.g., {shipping, delivery}
- Opinion-opinion pair, e.g., {quick, fast}
Aspect Extraction through Lifelong Recommendation

- AER (Aspect Extraction based on Recommendations) (Liu et al., 2016)

- Based on double propagation (Qiu et al., 2011)
  - Using syntactic relations
  - Detecting new aspects using known opinion words
  - Identifying new opinion words using known aspects
Two types of Recomm. in AER

- Similarity-based recommendation
  - Word2vec
  - Trained on a large corpus of 5.8 million reviews

- Aspect associations based recommendation
  - Association rule mining
  - Example: picture, display → video, purchase
Lifelong graph labeling for SA (Shu et al., 2016)

- **Problem:** opinion target labeling
  - Separating entities and aspects
  - Example: “Although the engine is slightly weak, this car is great.” **Entity:** car; **Aspect:** engine

- Suitable for lifelong learning
  - Similar usage or expression across domains
Lifelong graph labeling for SA
(Shu et al., 2016)

* Some words can be aspects in some domains, but entities in other domains
  - Battery is an **aspect** in “Camera”, “Laptop”, “Cellphone”
  - Battery is an entity in product “Battery”
LML knowledge base

- Type modifiers
  - E.g., in “this camera”, type of “camera” is entity

- Relation modifiers
  - E.g., in “the camera’s battery”, “camera” indicates an entity-aspect modifier for “battery”

- Predicted labels from past domains
Outline

- A motivating example
- What is lifelong machine learning?
- Related learning tasks
- Lifelong supervised learning
- Semi-supervised never-ending learning
- Lifelong unsupervised learning
- Lifelong reinforcement learning
- Summary
Reinforcement Learning

- An agent learns actions through trial and error interactions with a dynamic environment
- The agent gets reward/penalty after each action
- Each action changes the state of the environment
- The agent usually needs a large amount of quality experience (cost is high)
Lifelong Reinforcement Learning (LRL)

- Utilize the experience accumulated from other tasks
- Learn faster in a new task with fewer interactions
- Particularly useful in high-dimensional control problems
Example LRL Works

- Lifelong robot learning with knowledge memorization (Thrun and Mitchell 1995)
- Treating each environment as a task (Tanaka and Yamamura 1997)
- Hierarchical Bayesian approach (Wilson et al., 2007)
- Policy Gradient Efficient Lifelong Learning Algorithm (PG-ELLA) (Bou Ammar et al., 2014)
Outline

- A motivating example
- What is lifelong machine learning?
- Related learning tasks
- Lifelong supervised learning
- Semi-supervised never-ending learning
- Lifelong unsupervised learning
- Lifelong reinforcement learning
- Summary
Summary

- This tutorial gave an introduction to LML
  - By no means exhaustive

- **Existing LML research is still in its infancy**
  - The understanding of LML is very limited
  - Current research mainly focuses on
    - Only one type of tasks in a system

- **LML needs big data** – to learn a large amount of reliable knowledge of different types.
  - Little knowledge is not very useful
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)


Reference (3)


Reference (5)


