Learning a prior for lifelong visual object categorization

Royer Amélie

September 4, 2014

ENS rennes

IST AUSTRIA
Institute of Science and Technology

Classification

Classifier f:  $\mapsto$ Classes'score : $\begin{cases} Kit\ \ Fox? \\ Chair? \\ Cat? \\ Cowboy\ \ Hat? \end{cases}$

Classification

Classifier f:  $\mapsto$ Classes'score : $\begin{cases} Kit\ Fox? \\ Chair? \\ Cat? \\ Cowboy\ Hat? \end{cases}$

Usual Classification in real-life

- **Classifier**: Immediate visual information.
- **Human Being**: Infer additional knowledge from context.

On realistic sequences, how do we learn a context and use it?

Introducing Example

Wood Rabbit

Introducing Example

Wood Rabbit

Angora Rabbit

Introducing Example


Wood Rabbit


Watch

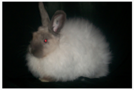

Angora Rabbit

Introducing Example

Wood Rabbit

Watch

Angora Rabbit



Angora Rabbit

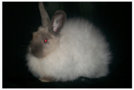
Introducing Example

Wood Rabbit

Watch

Watch

Angora Rabbit



Angora Rabbit

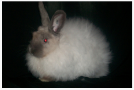
Introducing Example

Wood Rabbit

Watch

Watch

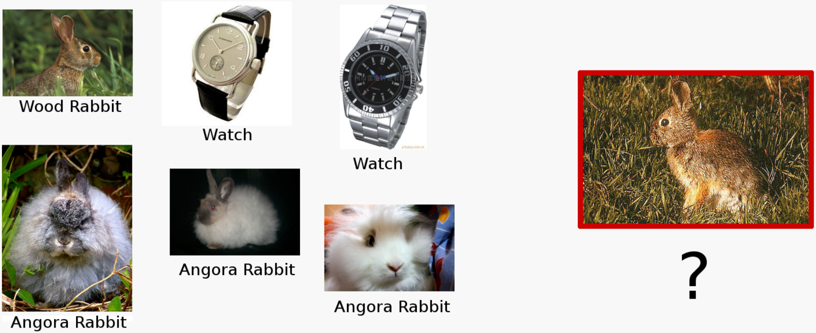
Angora Rabbit



Angora Rabbit

Angora Rabbit

Introducing Example



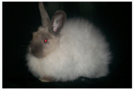
Introducing Example


Wood Rabbit

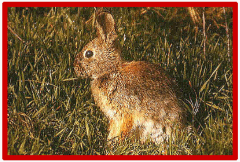

Watch


Watch


Angora Rabbit


Angora Rabbit


Angora Rabbit



Classifier predicts: Badger

With Context predicts: Wood Rabbit

Problem Formulation

Problem

- $\mathcal{X}$ (images), $\mathcal{Y}$ (classes)
- An initial classifier $f : \mathcal{X} \rightarrow R^{|\mathcal{Y}|}$
- A "realistic" sequence of queries $S = (x_i, y_i)_i \in (\mathcal{X}, \mathcal{Y})^N$

Problem Formulation

Problem

- $\mathcal{X}$ (images), $\mathcal{Y}$ (classes)
- An initial classifier $f : \mathcal{X} \rightarrow R^{|\mathcal{Y}|}$
- A "realistic" sequence of queries $S = (x_i, y_i)_i \in (\mathcal{X}, \mathcal{Y})^N$

Goals

1. Learn the context of S;
2. Combine it with the classifier f.

$\boxed{\Longrightarrow \text{ context-sensitive classifier } g}$

Online Learning

For (x_i, y_i) in the sequence S

1. g predicts a class $\hat{y}_i$;

Online Learning

`For` (x_i, y_i) `in the sequence` S

1. g predicts a class $\hat{y}_i$;
2. Receive feedback:

(Fully Supervised)
Receive the correct class y_i

(Reinforcement)
Receive the boolean information $(y_i == \hat{y}_i)$

(Unsupervised)
No Feedback.

Online Learning

`For` (x_i, y_i) `in the sequence` S

1. g predicts a class $\hat{y}_i$;
2. Receive feedback:

(Fully Supervised)
Receive the correct class y_i

(Reinforcement)
Receive the boolean information $(y_i == \hat{y}_i)$

(Unsupervised)
No Feedback.

3. g updates its knowledge of the context.

A probabilistic approach

Modelling a Context

$\boxed{\textbf{Context model} = \text{a probability distribution over the classes}}$

$$\theta_y(\mathbf{y}_0^{n-1}) = \mathbb{P}(y \mid \mathbf{y}_0^{n-1})$$

A probabilistic approach

Modelling a Context

Context model = a probability distribution over the classes

$$\theta_y(\mathbf{y}_0^{n-1}) = \mathbb{P}(y \mid \mathbf{y}_0^{n-1})$$

Domain adaptation

Source (Training time)($\mathbb{P}_s$): Uniform distribution of the queries.
Target (Testing time)($\mathbb{P}_t$): Unknown "context" of the sequence S.

A probabilistic approach

Modelling a Context

$\boxed{\textbf{Context model} = \text{a probability distribution over the classes}}$

$$\theta_y(\mathbf{y}_0^{n-1}) = \mathbb{P}(y \mid \mathbf{y}_0^{n-1})$$

Domain adaptation

Source (Training time)($\mathbb{P}_s$): Uniform distribution of the queries.
Target (Testing time)($\mathbb{P}_t$): Unknown "context" of the sequence S.

$$\boxed{\theta \sim \mathbb{P}_t}$$

Combining with the classifier

Combination

1. Domain adaptation (prior probability shift)
2. Bayes' rule in "source" and "target" settings.

Combining with the classifier

Combination

1. Domain adaptation (prior probability shift)
2. Bayes' rule in "source" and "target" settings.

$\Longrightarrow$

$$\boxed{g_y(x) = \mathbb{P}_t(y|x_n) \propto f_y(x_n) \times \theta_y(\mathbf{y}_0^{n-1})} \tag{1}$$

Combining with the classifier

Combination

1. Domain adaptation (prior probability shift)
2. Bayes' rule in "source" and "target" settings.

$\Longrightarrow$

$$\boxed{g_y(x) = \mathbb{P}_t(y|x_n) \propto f_y(x_n) \times \theta_y(\mathbf{y}_0^{n-1})} \tag{1}$$

$$\boxed{g_y(x) = \text{“score of } y \text{ given by } f\text{”} \times \text{“probability of } y \text{ in context } \theta\text{”}} \tag{2}$$

Using past frequencies

Fully-supervised: Multinomial Model

$$\theta_y(\mathbf{y}_0^{n-1}) = \mathbb{P}(y|\mathbf{y}_0^{n-1}) \triangleq \frac{w_{n-1}(y) + \varepsilon}{n + \varepsilon|\mathcal{Y}|}$$

- w_n : classes' counts up to round n
- ε smoothing term

Using past frequencies

Fully-supervised: Multinomial Model

$$\theta_y(\mathbf{y}_0^{n-1}) = \mathbb{P}(y|\mathbf{y}_0^{n-1}) \triangleq \frac{w_{n-1}(y) + \varepsilon}{n + \varepsilon|\mathcal{Y}|}$$

Update rule

`receive` true label y_n

$w_n(y_n) = w_{n-1}(y_n) + 1$
$w_n(y) = w_{n-1}(y), otherwise$

A more general approach

Confidence weight

1. classifier f : a score for each class (visual information)
2. context model : a "confidence weight" for each class

$$\Longrightarrow \textbf{Context} = \text{weight vector } w$$

$$g(x) = w \odot f(x)$$

($\odot$: component-wise multiplication).

A more general approach

Confidence weight

1. classifier f : a score for each class (visual information)
2. context model : a "confidence weight" for each class

$$\Longrightarrow \textbf{Context} = \text{weight vector } w$$

$$g(x) = w \odot f(x)$$

($\odot$: component-wise multiplication).

Weight vector ?

- Online learning;
- Multiplicative Update;
- Winnow.

Expert Weighting

Correcting the mistake

Mistake $\implies$ context must "correct" the initial classifier f.

Intuition of the update rule

If $y_n \neq \hat{y}_n$:

$$\forall y,\ w_n(y) \leftarrow \begin{cases} w_{n-1}(y) \times e^{\alpha(1-f_y(x_n))}, & \text{if } y = y_n \\ w_{n-1}(y) \times e^{-\alpha(1-f_y(x_n))}, & \text{if } y = \hat{y}_n \\ w_{n-1}(y), & \text{otherwise} \end{cases}$$

Realistic Sequence

What is “realistic” ?

Realistic: Semantical relation (e.g.: image sequence from human environment)

Realistic Sequence

What is “realistic” ?

Realistic: Semantical relation (e.g.: image sequence from human environment)

TXT database

- Browsing english books
- Retrieving labels

Realistic Sequence

What is “realistic” ?

Realistic: Semantical relation (e.g.: image sequence from human environment)

TXT database

- Browsing english books
- Retrieving labels

KS[1]and MDS[2]database

- Hierarchical Distance (ImageNet)
- 2D Projection
- Random Walk

[1] **NIPS2008˙0552**.

[2] **cox˙multidimensional˙2001**.

KS Example

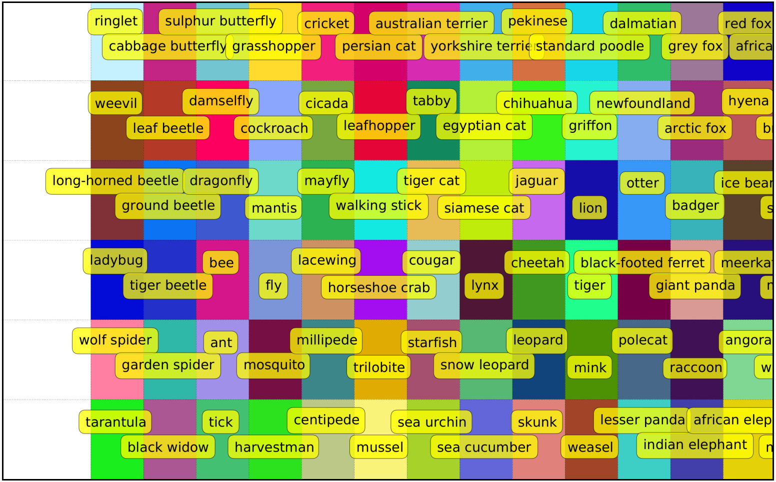


Figure : KS grid example

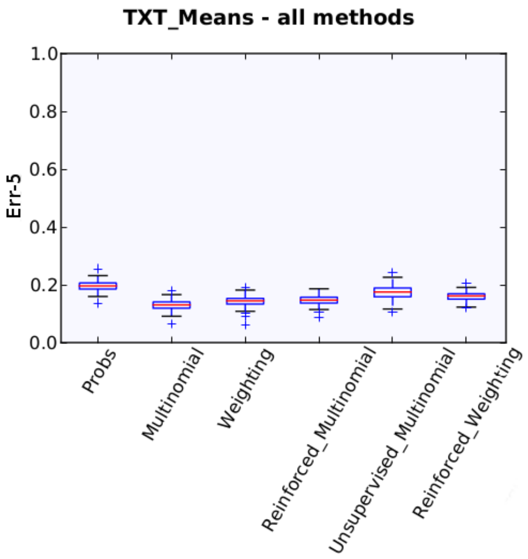
Experiments

Experimental Settings

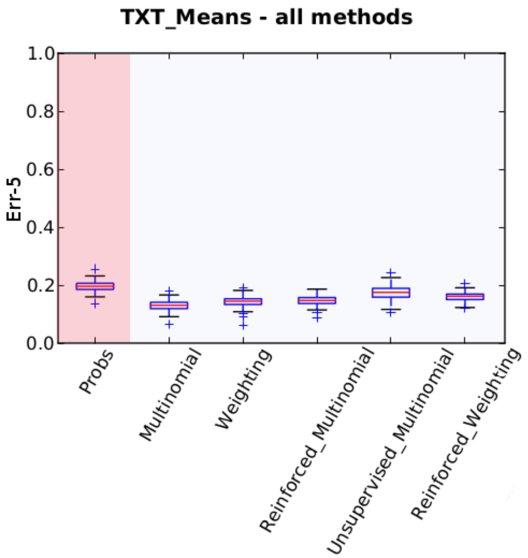
f = CCV classifier (convolutional neural network)

- 10 x 5000 `random` label sequences
- 100 `TXT` sequences
- 100 x 1500 `KS` sequences
- 100 x 1500 `MDS` sequences

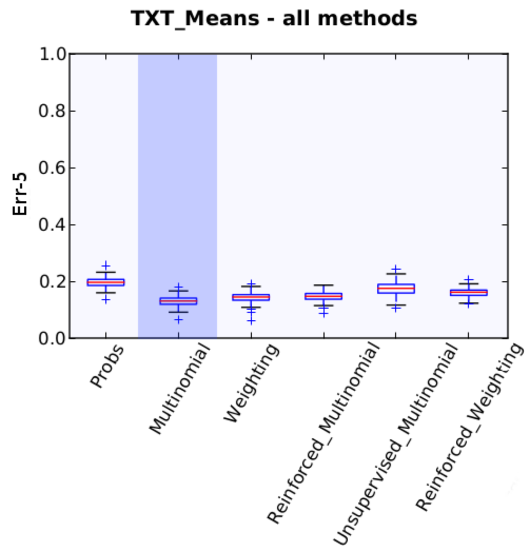
TXT Experiments



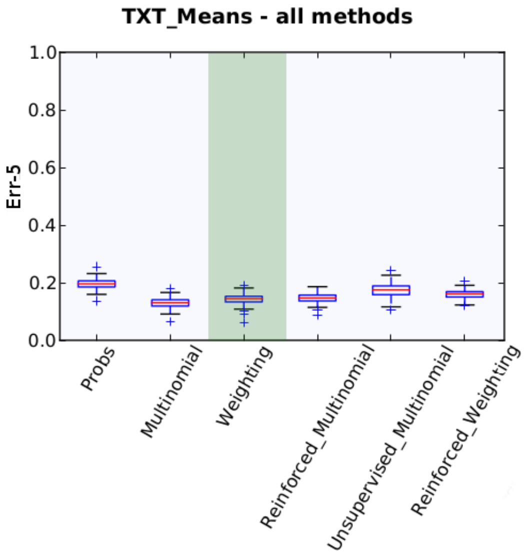
TXT Experiments



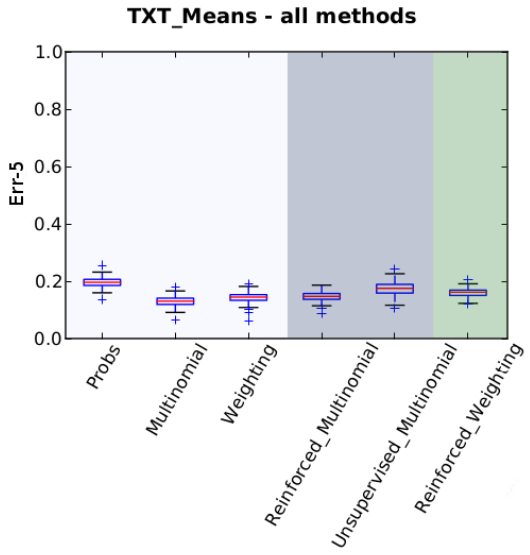
TXT Experiments



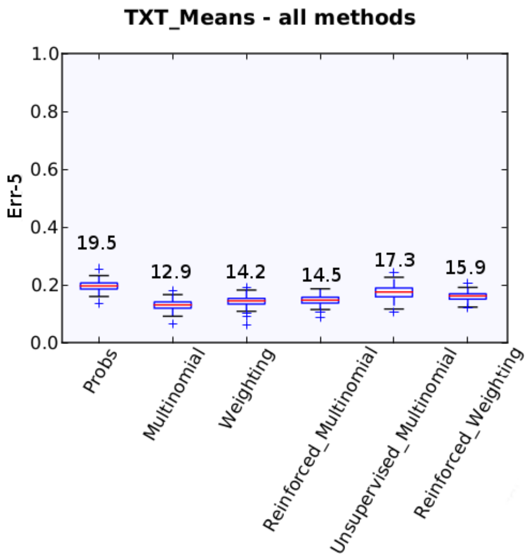
TXT Experiments



TXT Experiments



TXT Experiments



Conclusion

Conclusion

- Context knowledge improves classification accuracy on semantically ordered sequence
- Best method so far = Multinomial model
- Reinforcement and Unsupervised perform well when f is accurate

Conclusion

Conclusion

- Context knowledge improves classification accuracy on semantically ordered sequence
- Best method so far = Multinomial model
- Reinforcement and Unsupervised perform well when f is accurate

Future Works

- Theoretical Analysis
- More specific structures of sequences ?
- More specific context modelling methods ?

Conclusion

Conclusion

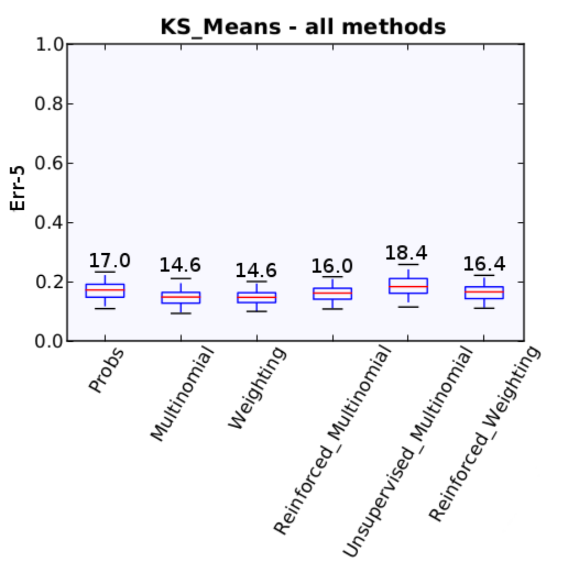
- Context knowledge improves classification accuracy on semantically ordered sequence
- Best method so far = Multinomial model
- Reinforcement and Unsupervised perform well when f is accurate

Future Works

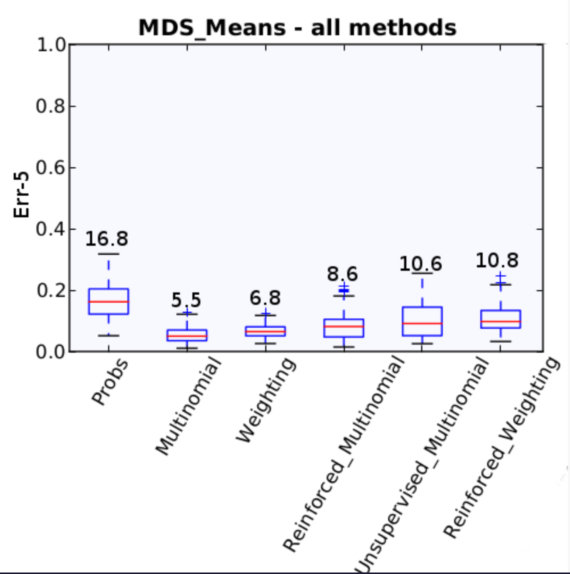
- Theoretical Analysis
- More specific structures of sequences ?
- More specific context modelling methods ?

Thanks for your attention

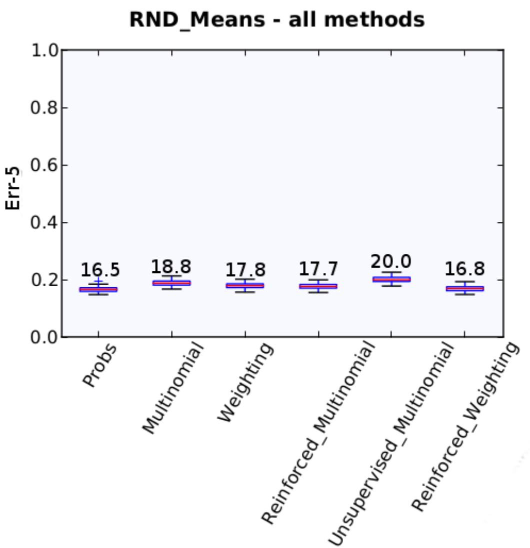
KS Results


KS_Means - all methods
Err-5
1.0
0.8
0.6
0.4
0.2
0.0
17.0
14.6
14.6
16.0
18.4
16.4
Probs
Multinomial
Weighting
Reinforced_Multinomial
Unsupervised_Multinomial
Reinforced_Weighting

MDS Results


MDS_Means - all methods
1.0
0.8
0.6
0.4
0.2
0.0
Err-5
16.8
5.5
6.8
8.6
10.6
10.8
Probs
Multinomial
Weighting
Reinforced_Multinomial
Unsupervised_Multinomial
Reinforced_Weighting

RND Experiments



Combining with the classifier

Combination

1. Domain adaptation (prior probability shift):
   - $\mathbb{P}_s(y) \neq \mathbb{P}_t(y)$
   - $\mathbb{P}_s(x|y) = \mathbb{P}_t(x|y)$

Combining with the classifier

Combination

1. Domain adaptation (prior probability shift):
   - $\mathbb{P}_s(y) \neq \mathbb{P}_t(y)$
   - $\mathbb{P}_s(x|y) = \mathbb{P}_t(x|y)$
2. Bayes' rule in "source" and "target" settings.

$$\forall y \in \mathcal{Y},\ f(x_n)_y = \mathbb{P}_s(y|x_n) = \frac{\mathbb{P}_s(x_n|y) \times \frac{1}{|\mathcal{Y}|}}{\mathbb{P}_s(x_n)} \tag{3}$$

$$\forall y \in \mathcal{Y},\ \mathbb{P}_t(y|x_n) = \frac{\mathbb{P}_t(x_n|y) \times \theta_y(\mathbf{y}_0^{n-1})}{\mathbb{P}_t(x_n)} \tag{4}$$

Combining with the classifier

Combination

1. Domain adaptation (prior probability shift):
   - $\mathbb{P}_s(y) \neq \mathbb{P}_t(y)$
   - $\mathbb{P}_s(x|y) = \mathbb{P}_t(x|y)$

Combining with the classifier

Combination

1. Domain adaptation (prior probability shift):
   - $\mathbb{P}_s(y) \neq \mathbb{P}_t(y)$
   - $\mathbb{P}_s(x|y) = \mathbb{P}_t(x|y)$
2. Bayes' rule in "source" and "target" settings.

$\Longrightarrow$

$$\boxed{g_y(x) = \mathbb{P}_t(y|x_n) \propto f_y(x_n) \times \theta_y(\mathbf{y}_0^{n-1})} \tag{5}$$

$$\boxed{g_y(x) = \text{"score of } y \text{ given by } f\text{"} \times \text{"probability of } y \text{ in context } \theta\text{"}} \tag{6}$$

Multinomial Model

Reinforcement

`receive` $y_n == \hat{y}_n$

If $y_n == \hat{y}_n$: Same rule
Else: $w_n(y) = w_{n-1}(y) + \frac{1}{|\mathcal{Y}|-1}$, if $y \neq \hat{y}_n$

Multinomial Model

Reinforcement

`receive` $y_n == \hat{y}_n$

If $y_n == \hat{y}_n$: Same rule
Else: $w_n(y) = w_{n-1}(y) + \frac{1}{|\mathcal{Y}|-1}$, if $y \neq \hat{y}_n$

Unsupervised

`receive` nothing

$w_n(\hat{y}_n) = w_{n-1}(\hat{y}_n) + 1$
$w_n(y) = w_{n-1}(y), otherwise$

Weighting Model

Reinforcement

`receive` $y_n == \hat{y}_n$

If $y_n == \hat{y}_n$: Positive update.

$$\forall y,\ w_n(y) \leftarrow \begin{cases} w_{n-1}(y) \times e^{\alpha(\mathbf{1}-\mathbf{f_y}(\mathbf{x_n}))}, & \text{if } y = y_n \\ w_{n-1}(y) & \end{cases}$$

Else: Negative update.

$$\forall y,\ w_n(y) \leftarrow \begin{cases} w_{n-1}(y) \times e^{-\alpha(\mathbf{1}-\mathbf{f_y}(\mathbf{x_n}))}, & \text{if } y = \hat{y}_n \\ w_{n-1}(y) & \end{cases}$$