Multi-Task Structured Prediction for Entity Analysis: Search Based Learning Algorithms

Chao Ma, †Janardhan Rao Doppa, Prasad Tadepalli, Hamed Shahbazi, Xiaoli Fern

Oregon State University
†Washington State University

Nov. 17th, 2017

ACML 2017

Entity Analysis in Language Processing

Many NLP tasks process mentions of entities – things, people, organizations, etc.

- **Named Entity Recognition**
- **Coreference Resolution**
- **Entity Linking**
- Semantic Role Labeling
- Entity Relation Extraction

......

We focus on three of them in this work

ACML 2017

Coreference Resolution

***i* = 1** **_i_ = 2**

He left [**Columbia**] in 1983 with a BA degree, ... after graduating from [**Columbia University**], he worked as a community organizer in Chicago…

Coreference Resolution

i = 1 i = 2

He left [**Columbia**] in 1983 with a BA degree, ... after graduating from [**Columbia University**], he worked as a community organizer in Chicago…

Coreference:

$y_i = \{1, 2 \ldots i\}$

co-referent link

co-referent link

$$\boldsymbol{y}_{\text{coref}} = (\text{Columbia}, \text{Columbia University}, \ldots)$$

ACML 2017

Coreference Resolution

$\boldsymbol{i} = \mathbf{1}$ $\boldsymbol{i} = \mathbf{2}$

He left [**Columbia**] in 1983 with a BA degree, ... after graduating from [**Columbia University**], he worked as a community organizer in Chicago…

co-referent link co-referent link

Columbia **Columbia University**

Coreference:

$y_{\mathrm{i}} = \{1, 2 \ldots i\}$

Left-linking Tree formulation for coreference resolution:

m_1 m_2 m_3 m_4 m_5 m_6 m_7

$y_{\text{coref}} = (\ ?\ ,\ ?\ ,\ ?\ ,\ ?\ ,\ ?\ ,\ ?\ ,\ ?\ )$

ACML 2017

Coreference Resolution

$i = 1$ $i = 2$

He left [**Columbia**] in 1983 with a BA degree, ... after graduating from [**Columbia University**], he worked as a community organizer in Chicago…

co-referent link

Columbia

co-referent link

Columbia University

Coreference:

$y_i = \{1, 2 \ldots i\}$

Left-linking Tree formulation for coreference resolution:

m_1 m_2 m_3 m_4 m_5 m_6 m_7

$$y_{coref} = (\ 1,\ ?,\ ?,\ ?,\ ?,\ ?,\ ?\ )$$

ACML 2017

Coreference Resolution

$i = 1$ $\quad$ $i = 2$

He left [**Columbia**] in 1983 with a BA degree, … after graduating from [**Columbia University**], he worked as a community organizer in Chicago…

co-referent link $\quad$ co-referent link

Columbia $\quad$ Columbia University

Coreference:

$y_i = \{1, 2 \dots i\}$

Left-linking Tree formulation for coreference resolution:

m_1 $\quad$ m_2 $\quad$ m_3 $\quad$ m_4 $\quad$ m_5 $\quad$ m_6 $\quad$ m_7

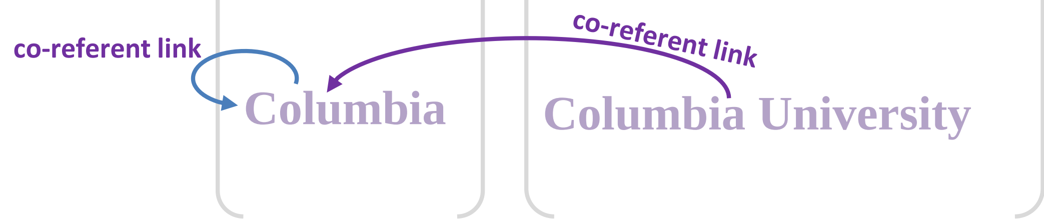
$$y_{\text{coref}} = (\ 1,\ 1,\ ?,\ ?,\ ?,\ ?,\ ?\ )$$

ACML 2017

Coreference Resolution

$i = 1$ $\quad$ $i = 2$

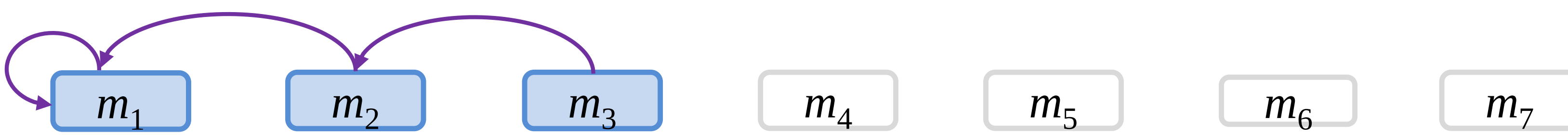
He left [**Columbia**] in 1983 with a BA degree, ... after graduating from [**Columbia University**], he worked as a community organizer in Chicago…



Coreference:

$y_i = \{1, 2 \ldots i\}$

Left-linking Tree formulation for coreference resolution:



$y_{coref} = (\ 1\ ,\ 1\ ,\ 2\ ,\ ?\ ,\ ?\ ,\ ?\ ,\ ?\ )$

Coreference Resolution

$i = 1$ $\quad$ $i = 2$

He left [**Columbia**] in 1983 with a BA degree, ... after graduating from [**Columbia University**], he worked as a community organizer in Chicago…

co-referent link

Columbia

co-referent link

Columbia University

Coreference:

$y_i = \{1, 2 \ldots i\}$

Left-linking Tree formulation for coreference resolution:

m_1 $\quad m_2$ $\quad m_3$ $\quad m_4$ $\quad m_5$ $\quad m_6$ $\quad m_7$

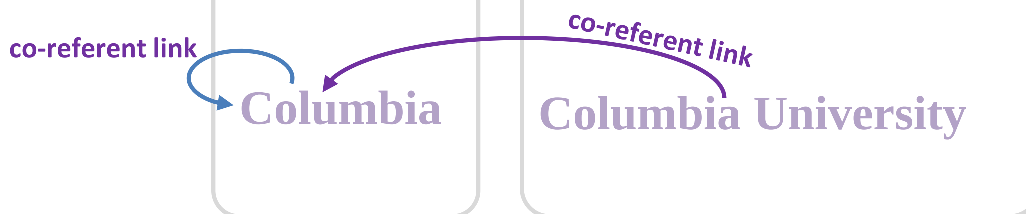
$y_{\text{coref}} = (\; 1 \;, \; 1 \;, \; 2 \;, \; 4 \;, \; ? \;, \; ? \;, \; ? \;)$

ACML 2017

Coreference Resolution

$i = 1$ $\quad$ $i = 2$

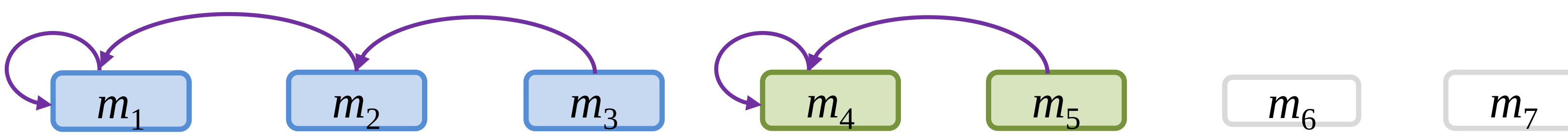
He left [**Columbia**] in 1983 with a BA degree, … after graduating from [**Columbia University**], he worked as a community organizer in Chicago…



Coreference:

$y_i = \{1, 2 \ldots i\}$

Left-linking Tree formulation for coreference resolution:



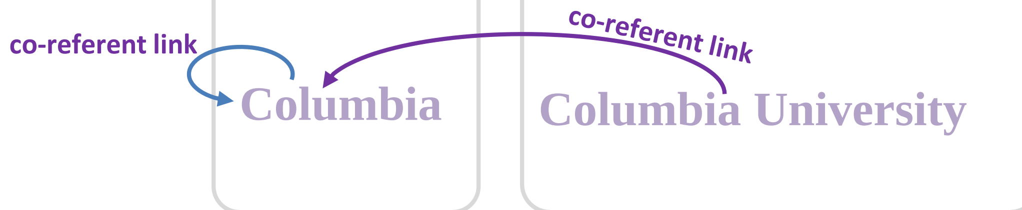
$y_{\text{coref}} = (\; 1, \; 1, \; 2, \; 4, \; 5, \; ?, \; ? \;)$

ACML 2017

Coreference Resolution

$\boldsymbol{i} = \mathbf{1}$ $\boldsymbol{i} = \mathbf{2}$

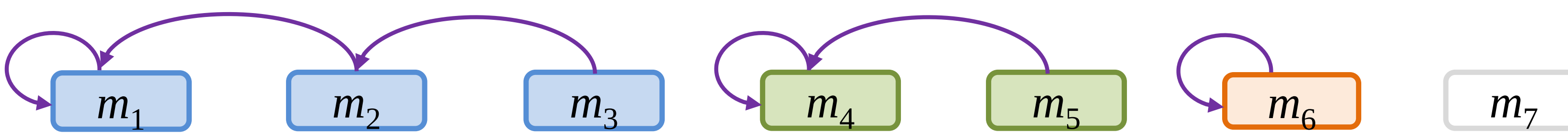
He left [**Columbia**] in 1983 with a BA degree, ... after graduating from [**Columbia University**], he worked as a community organizer in Chicago…



Coreference:

$y_i = \{1, 2 \ldots i\}$

Left-linking Tree formulation for coreference resolution:



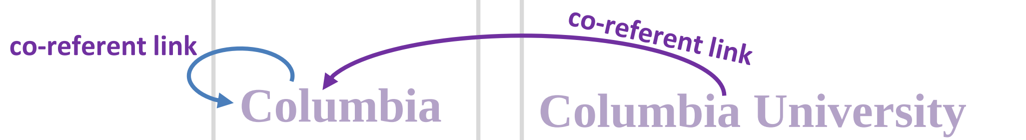
$$y_{\text{coref}} = (\ 1,\ 1,\ 2,\ 4,\ 5,\ 6,\ ?\ )$$

ACML 2017

Coreference Resolution

$i = 1$ $i = 2$

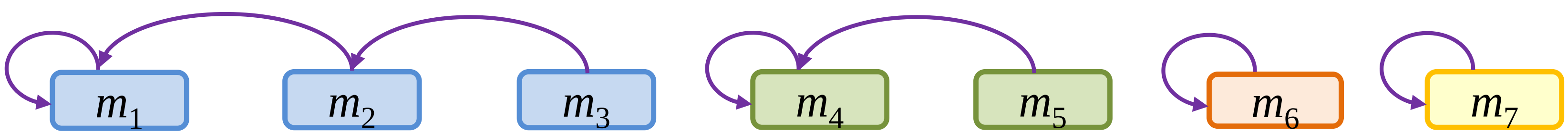
He left [**Columbia**] in 1983 with a BA degree, ... after graduating from [**Columbia University**], he worked as a community organizer in Chicago…



Coreference:

$$y_i = \{1, 2 \ldots i\}$$

Left-linking Tree formulation for coreference resolution:



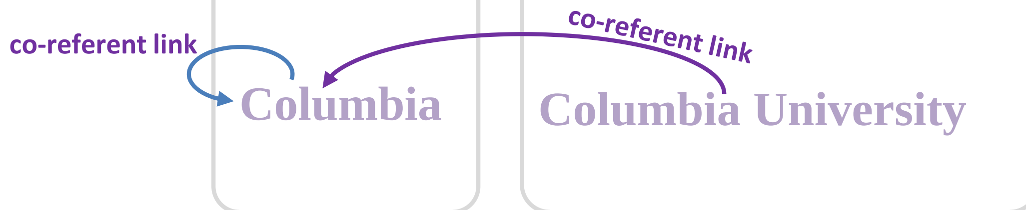
$$y_{\text{coref}} = (\ 1,\ 1,\ 2,\ 4,\ 5,\ 6,\ 7\ )$$

ACML 2017

Coreference Resolution

$i = 1$ $i = 2$

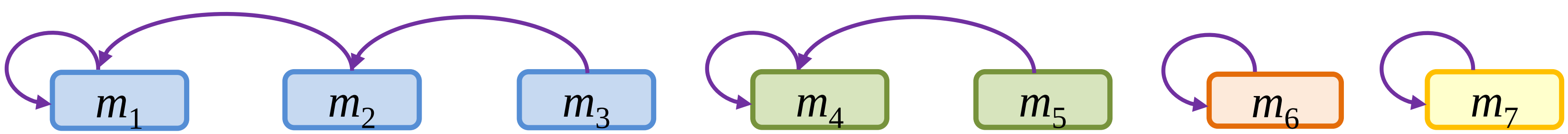
He left [**Columbia**] in 1983 with a BA degree, ... after graduating from [**Columbia University**], he worked as a community organizer in Chicago…



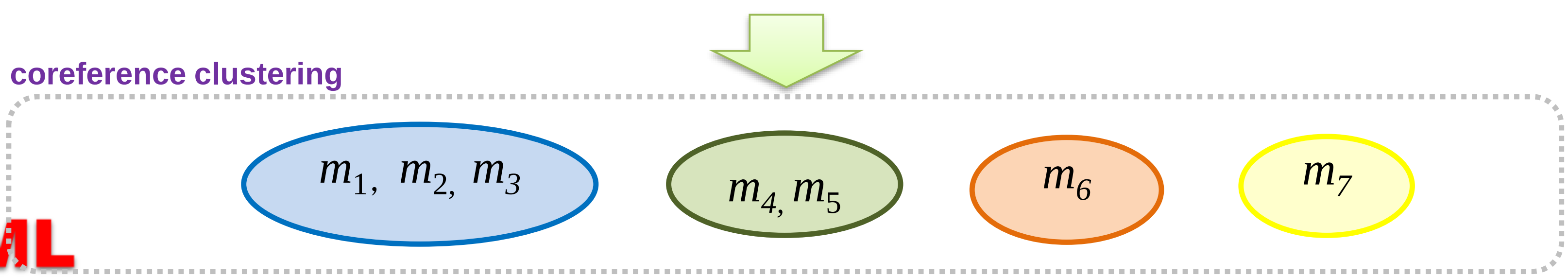
Coreference:

$y_i = \{1, 2 \ldots i\}$

Left-linking Tree formulation for coreference resolution:



$y_{\text{coref}} = (1, 1, 2, 4, 5, 6, 7)$



Coreference Resolution

$i = 1$ $i = 2$

He left [**Columbia**] in 1983 with a BA degree, ... after graduating from [**Columbia University**], he worked as a community organizer in Chicago…

co-referent link

co-referent link

Coreference: $y_{\text{coref}} = $ (Columbia , Columbia University , …)

$y_{\mathrm{i}} = \{1, 2 \ldots i\}$

ACML 2017

Named Entity Recognition

$i = 1$ $i = 2$

He left [**Columbia**] in 1983 with a BA degree, ... after graduating from [**Columbia University**], he worked as a community organizer in Chicago…

co-referent link co-referent link

Coreference: $y_{\text{coref}} =$ (Columbia , Columbia University , …)

$y_i = \{1, 2 \ldots i\}$

Named Entity Recognition : $y_{\text{ner}} =$ (ORG, ORG, …)

$y_i = \{$ORG, PER, GPE, LOC, FAC, VEL, WEA$\}$

Entity Linking

$i = 1$ $i = 2$

He left [**Columbia**] in 1983 with a BA degree, … after graduating from [**Columbia University**], he worked as a community organizer in Chicago…

co-referent link

co-referent link

Coreference: $y_{\text{coref}} =$ (Columbia , Columbia University , …)

$y_i = \{1, 2 \ldots i\}$

Named Entity Recognition : $y_{\text{ner}} =$ (ORG, ORG, …)

$y_i = \{$ORG, PER, GPE, LOC, FAC, VEL, WEA$\}$

Entity Linking: $y_{\text{link}} =$ (https://en.wikipedia.org/wiki/Columbia_University , https://en.wikipedia.org/wiki/Columbia_University , …)

$y_i = \{$
https://en.wikipedia.org/wiki/Columbia_University,
https://en.wikipedia.org/wiki/Columbia_District,
https://en.wikipedia.org/wiki/Columbia,_British_Columbia,
https://en.wikipedia.org/wiki/Columbia_College,_Columbia_University,
.
$\}$

Single Task Structured Prediction

Typical (Single-Task) Structured Prediction:

x → $f: \mathcal{X}, \mathcal{Y} \longrightarrow \mathcal{R}$ → y

Input

Scoring function

Output

Learning

$$f(x, y) = w \cdot \underbrace{\phi(x, y)}_{\text{Feature Vector}}$$

Inference

$$\hat{y} = \underbrace{\operatorname*{argmax}_{y} f(x, y)}_{\text{Intractable in most cases}}$$

Candidate Methods:

- Graphical models
- Structured Perceptron
- Structured SVM

……

Candidate Methods:

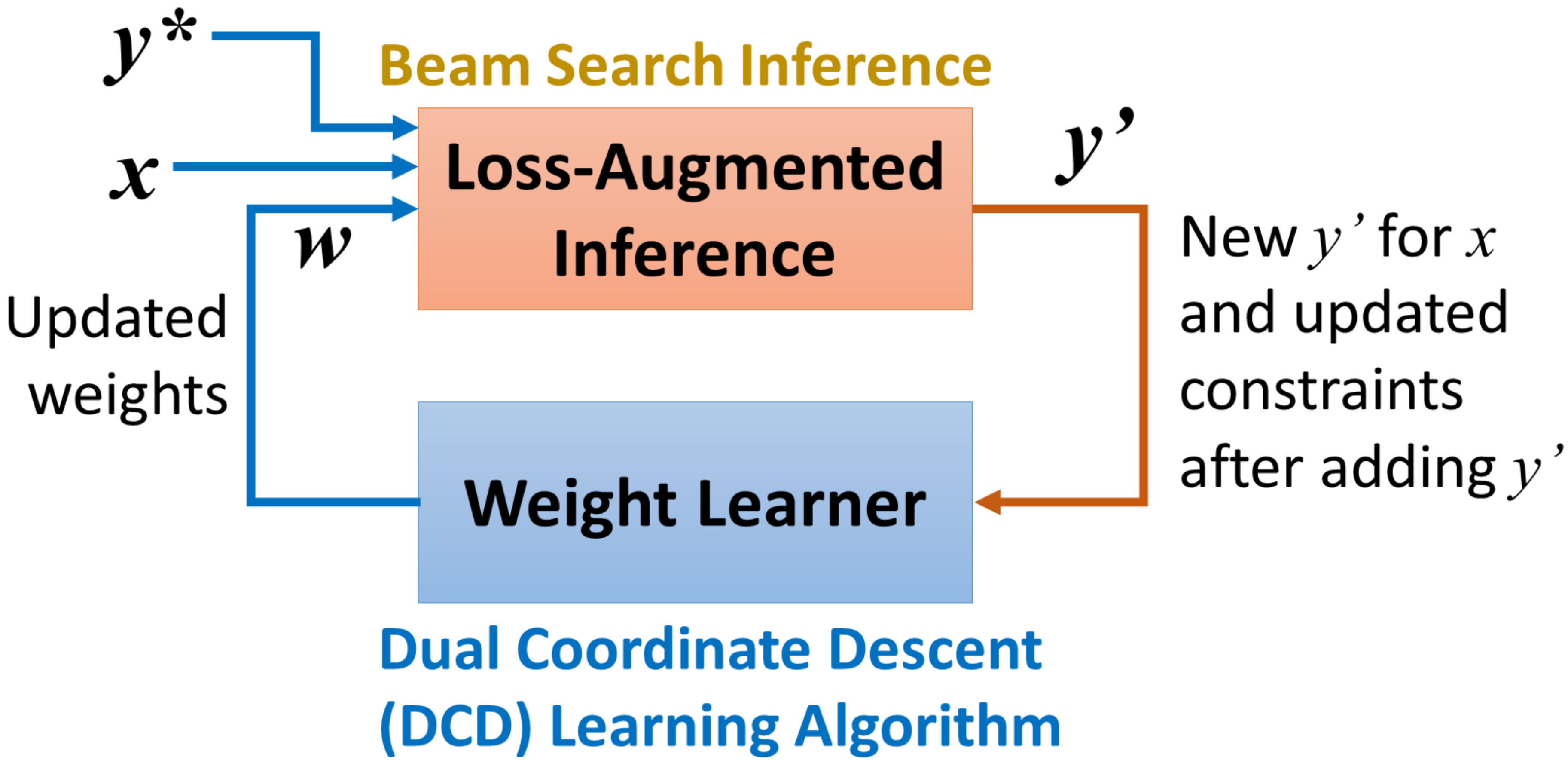
- Belief Propagation
- Integer Linear Programming (ILP)
- Beam Search

……

This Work

ACML 2017

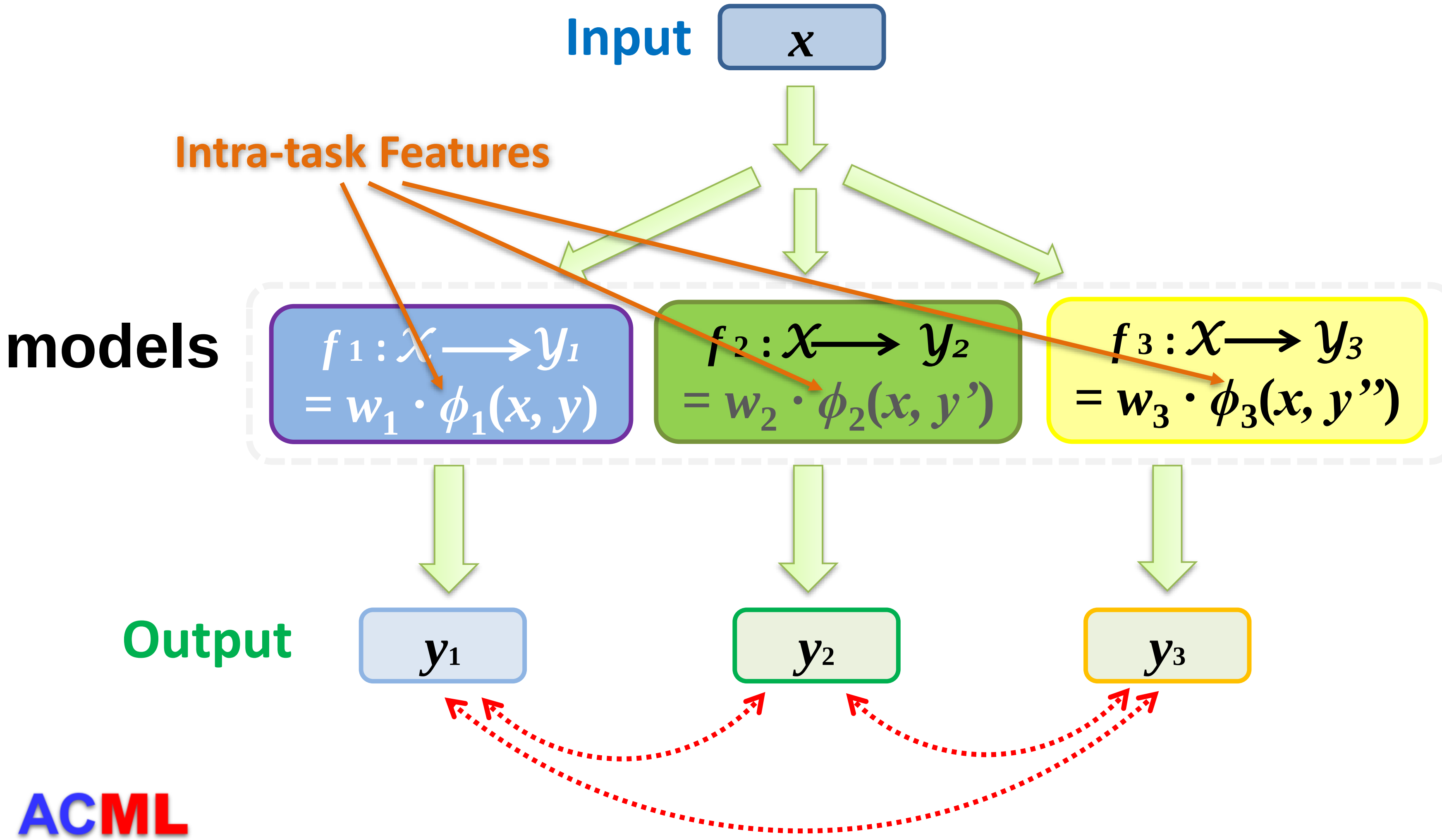
Structured SVM Learning with Search-based Inference



ACML 2017

Multi-Task Structured Prediction

Multi-Task Structured Prediction (MTSP):

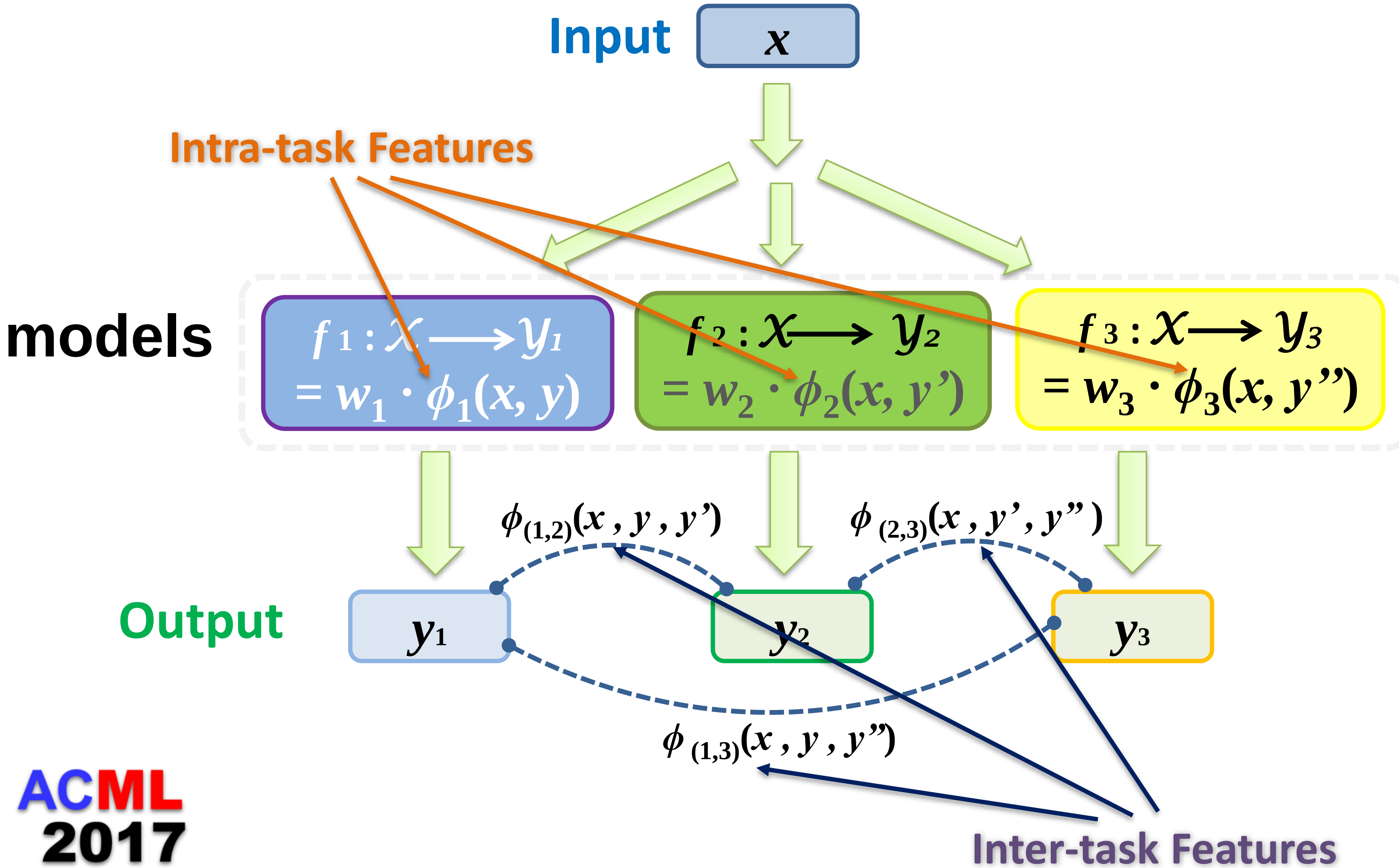


ACML 2017

- *How to exploit the interdependencies between tasks?*

Multi-Task Structured Prediction

Introduce Inter-task Features:



ACML 2017

Pipeline Architecture

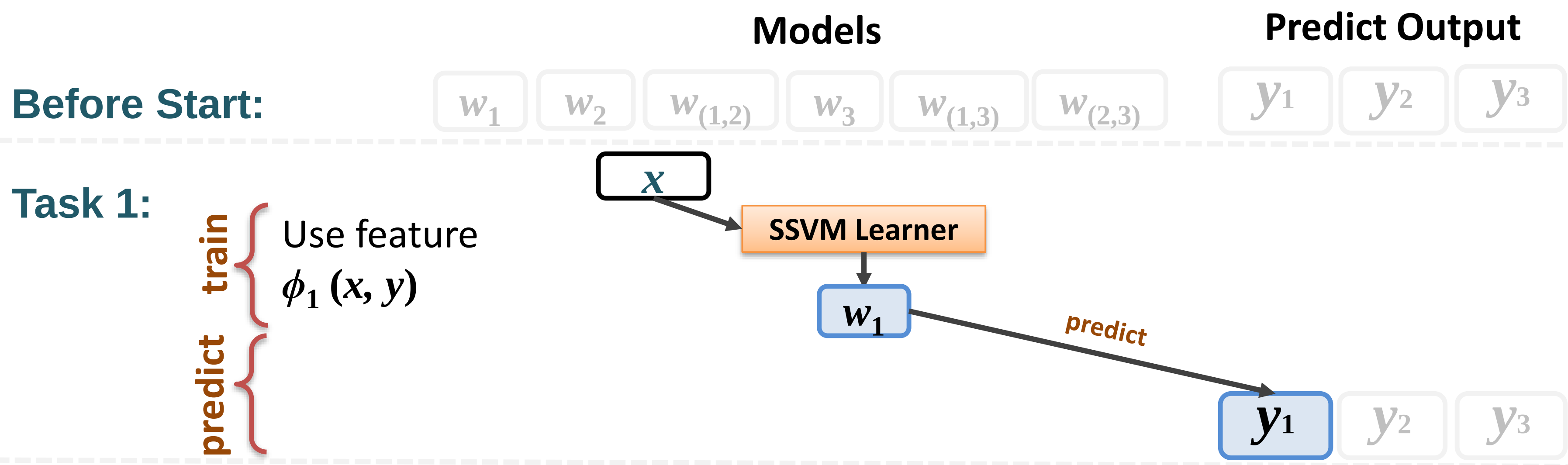
Learning k (= 3) independent models, one after another;

	Models						**Predict Output**		
Before Start:	w_1	w_2	$w_{(1,2)}$	w_3	$w_{(1,3)}$	$w_{(2,3)}$	y_1	y_2	y_3

Define a order: Task 1 → Task 2 → Task 3

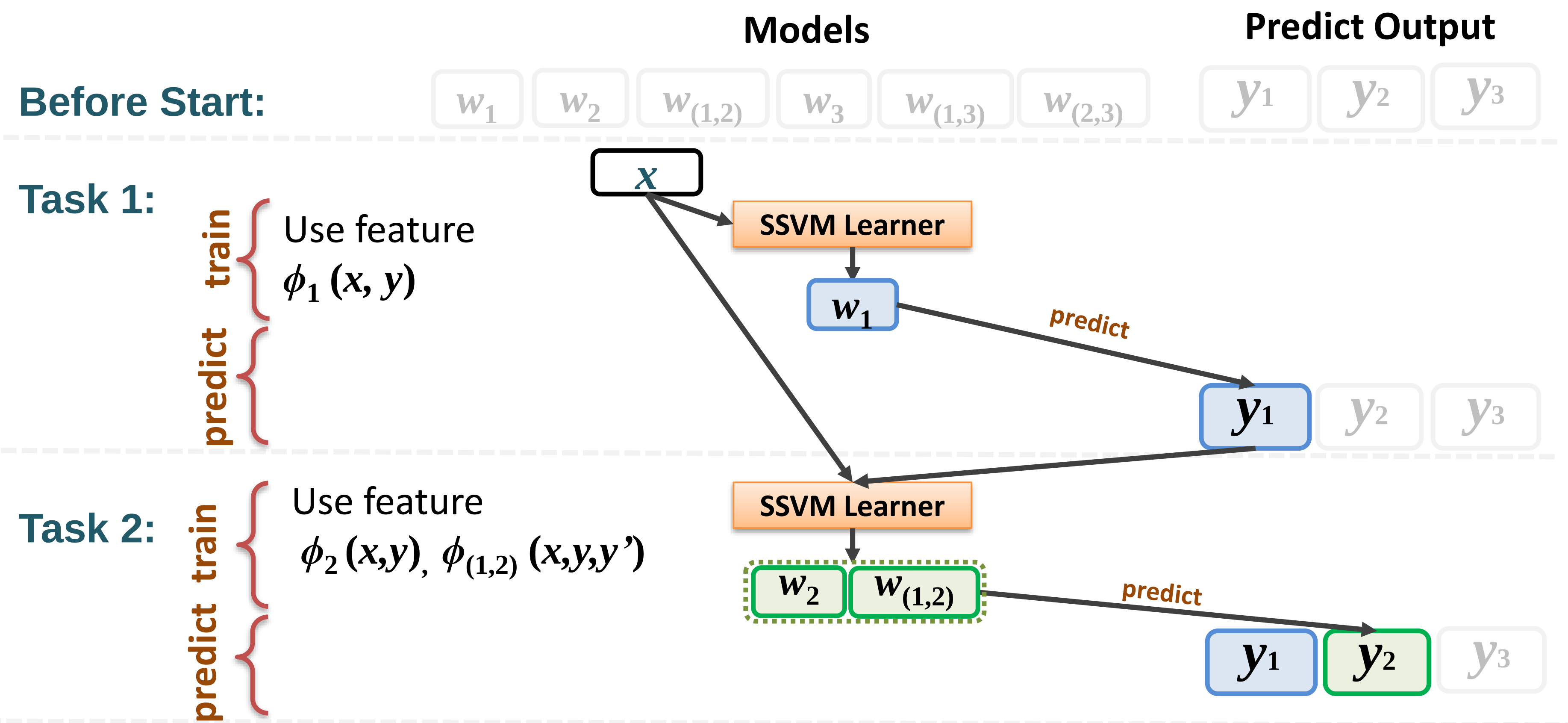
Pipeline Architecture

Learning k (= 3) independent models, one after another;



Pipeline Architecture

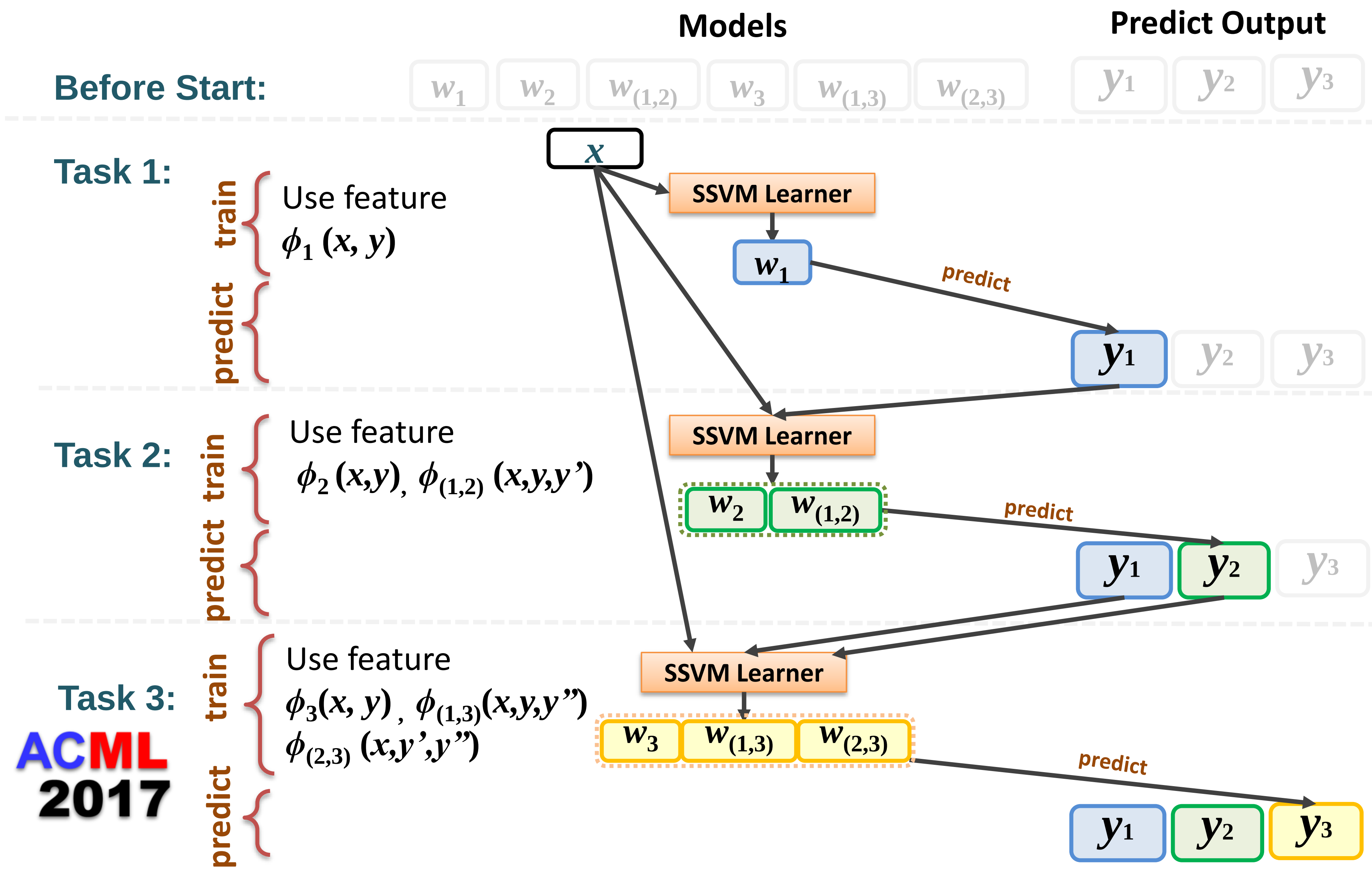
Learning k (= 3) independent models, one after another;



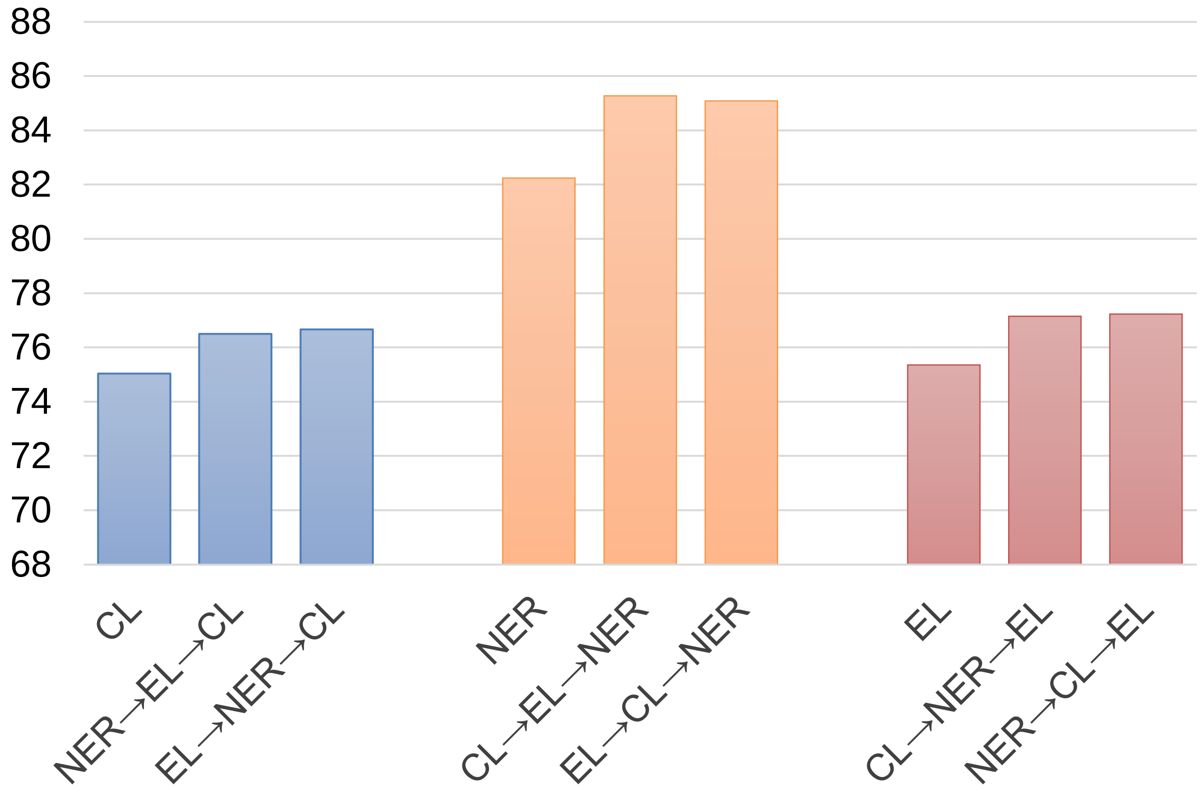
ACML 2017

Pipeline Architecture

Learning k (= 3) independent models, one after another;



Pipeline Performance Depends on Task Order



✓ Each group of bars represents one task. In each group, we show the accuracy when the task is placed at first (1st bar), or at last (2nd and 3rd bar).

- ❑ The task performs better when it is placed last in order.
- ❑ There is **no** ordering that allows the pipeline to reach peak performance on all the three tasks.

ACML 2017

Joint Architecture

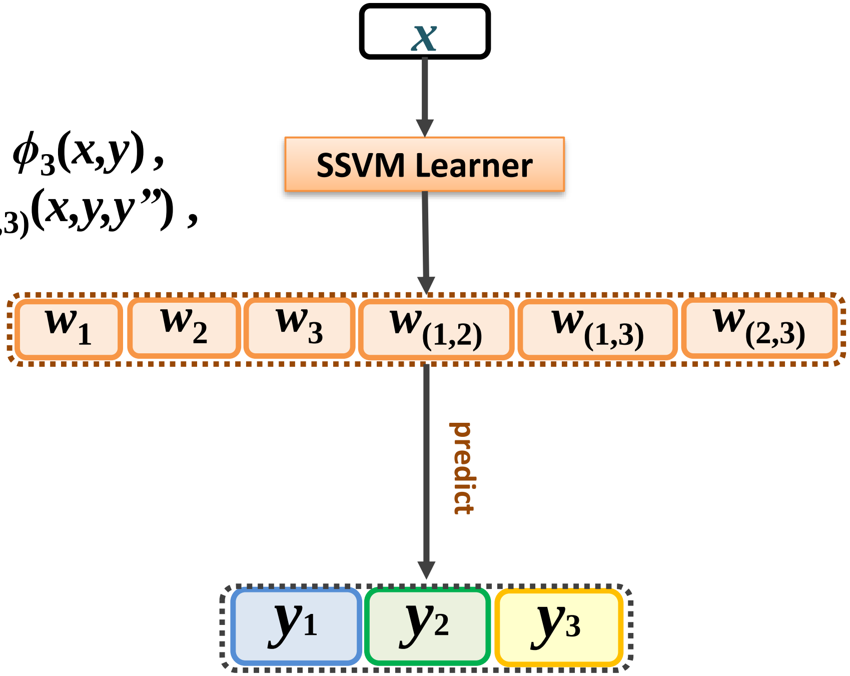
Task 1 & 2 & 3:

train

Use all features

$\phi_1(x,y)$, $\phi_2(x,y)$, $\phi_3(x,y)$, $\phi_{(1,2)}(x,y,y')$, $\phi_{(1,3)}(x,y,y'')$, $\phi_{(2,3)}(x,y',y'')$

predict



$$\phi = \phi_1(x,y) \circ \phi_2(x,y) \circ \phi_3(x,y) \circ \phi_{(1,2)}(x,y,y') \circ \phi_{(1,3)}(x,y,y'') \circ \phi_{(2,3)}(x,y',y'')$$

Vector concatenation

Big Problem: Huge branching factor for search

Pruning

A pruner is a classifier to prune the domain of each variable using state features.

Score-agnostic Pruning

Training Data —(Pruner training & predicting)→ Pruned Data —(SSVM Learner)→ Cost function —(testing)→ Testing Results

- Can accelerate the training time;
- May or may not improve the testing accuracy;

Score-sensitive Pruning

Training Data —(SSVM Learner)→ Cost function —(Pruner training & predicting)→ Testing Results

- Can improve the testing accuracy;
- No training speedup, but evaluation does speed up.

Cyclic Architecture

Pipeline architecture

Task 1 → Task 2 → Task 3

Connect the tail of pipeline to the head?

Cyclic Architecture

Unshared-Weight-Cyclic Training

Step 1: **Define a order: Task 1 → Task 2 → Task 3**

Step 2: **Predict initial outputs:** $\boxed{y_1}$ $\boxed{y_2}$ $\boxed{y_3}$

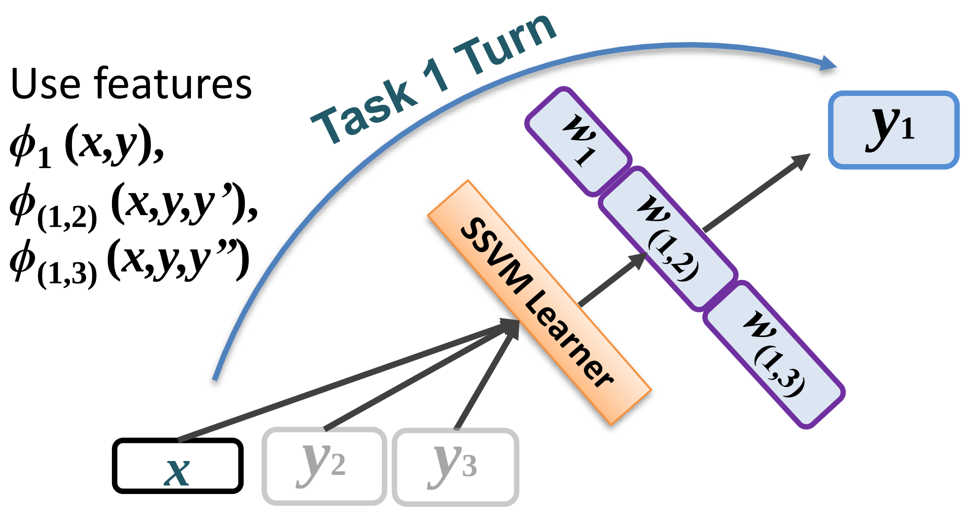
ACML 2017

Cyclic Architecture

Unshared-Weight-Cyclic Training

Step 1: Define a order: Task 1 → Task 2 → Task 3

Step 2: Predict initial outputs: y_1 y_2 y_3



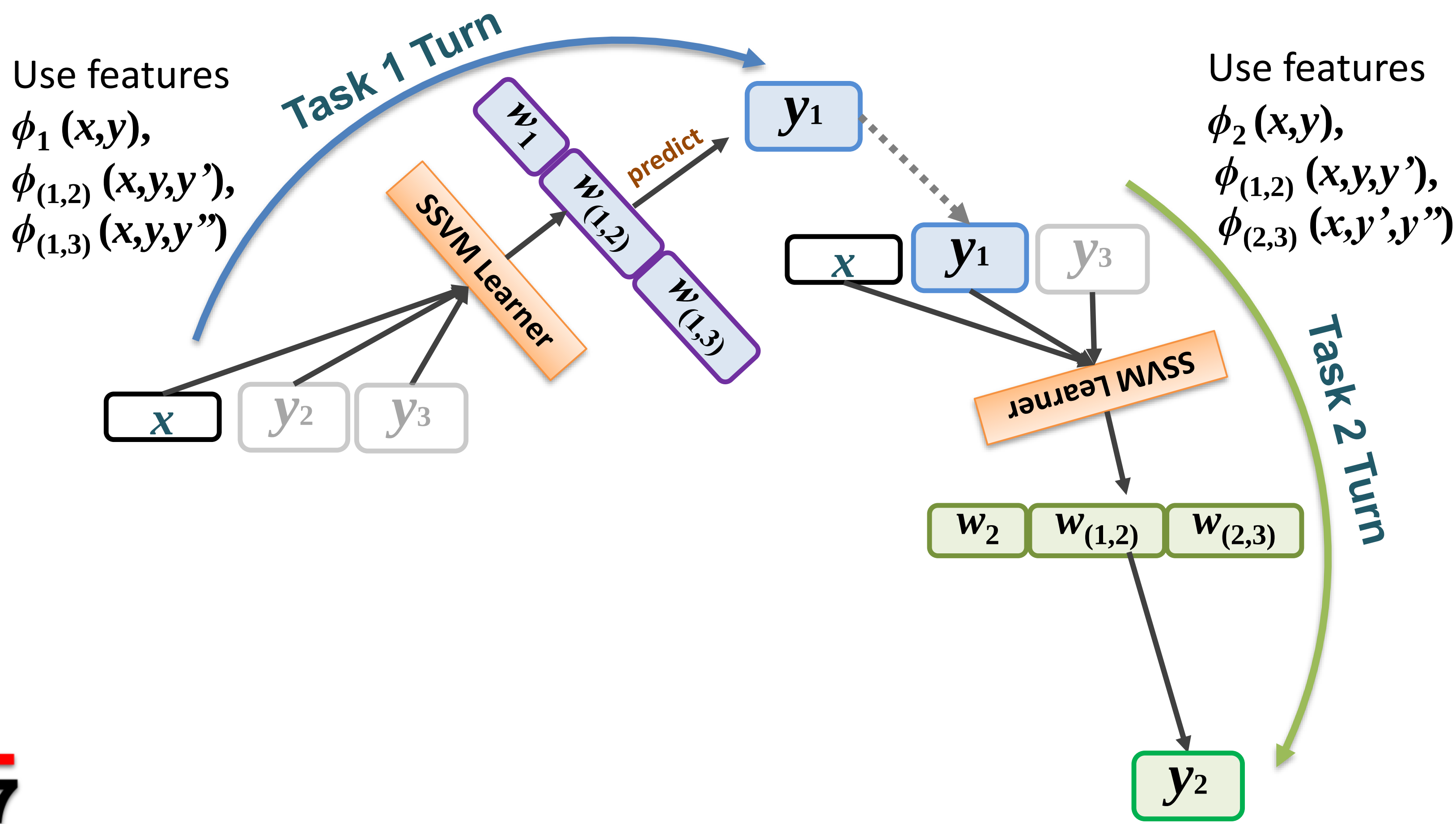
ACML 2017

Cyclic Architecture

Unshared-Weight-Cyclic Training

Step 1: Define a order: Task 1 → Task 2 → Task 3

Step 2: Predict initial outputs: y_1 y_2 y_3

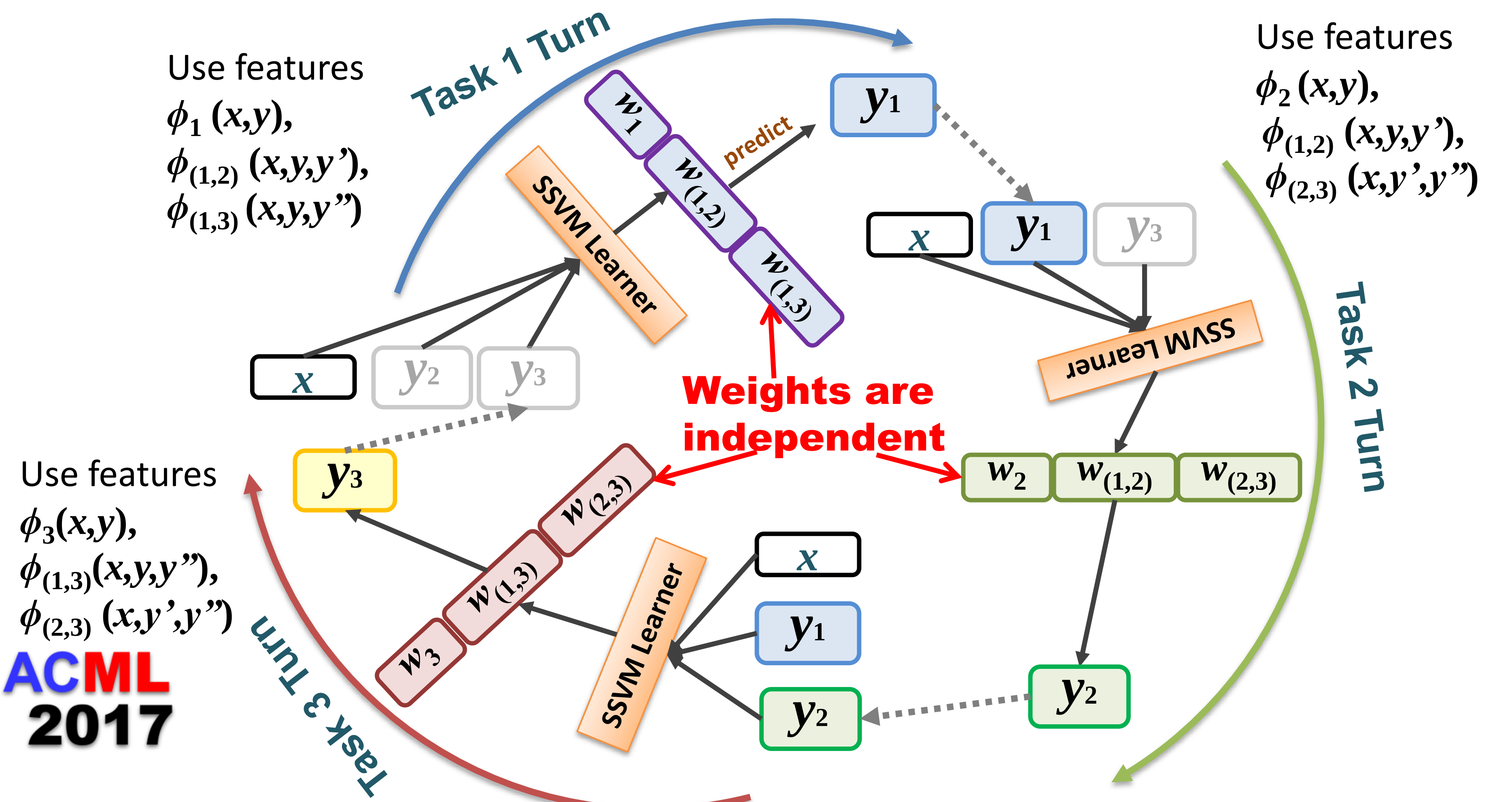


Cyclic Architecture

Unshared-Weight-Cyclic Training

Step 1: Define a order: Task 1 → Task 2 → Task 3

Step 2: Predict initial outputs: y_1 y_2 y_3



Cyclic Architecture

Shared-Weight-Cyclic Training

Step 1: Define a order: Task 1 → Task 2 → Task 3

Step 2: Predict initial outputs: y_1 y_2 y_3

x y_1 y_2 y_3

w_1 w_2 w_3 $w_{(1,2)}$ $w_{(1,3)}$ $w_{(2,3)}$

Cyclic Architecture

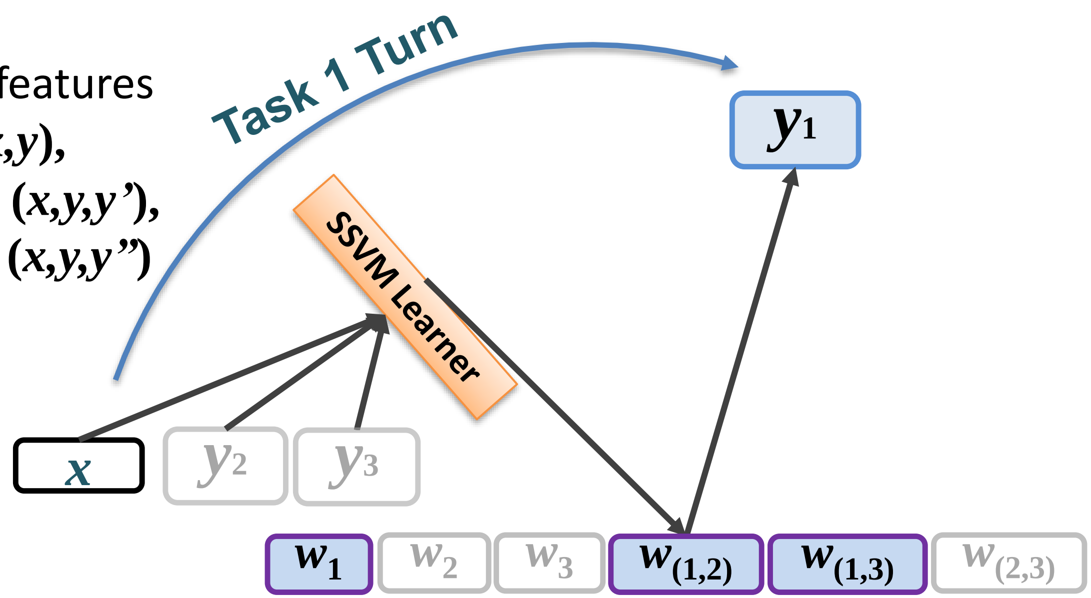
OSU Oregon State UNIVERSITY

Shared-Weight-Cyclic Training

Step 1: Define a order: Task 1 → Task 2 → Task 3

Step 2: Predict initial outputs: y_1 y_2 y_3

Use features
$\phi_1(x,y)$,
$\phi_{(1,2)}(x,y,y')$,
$\phi_{(1,3)}(x,y,y'')$



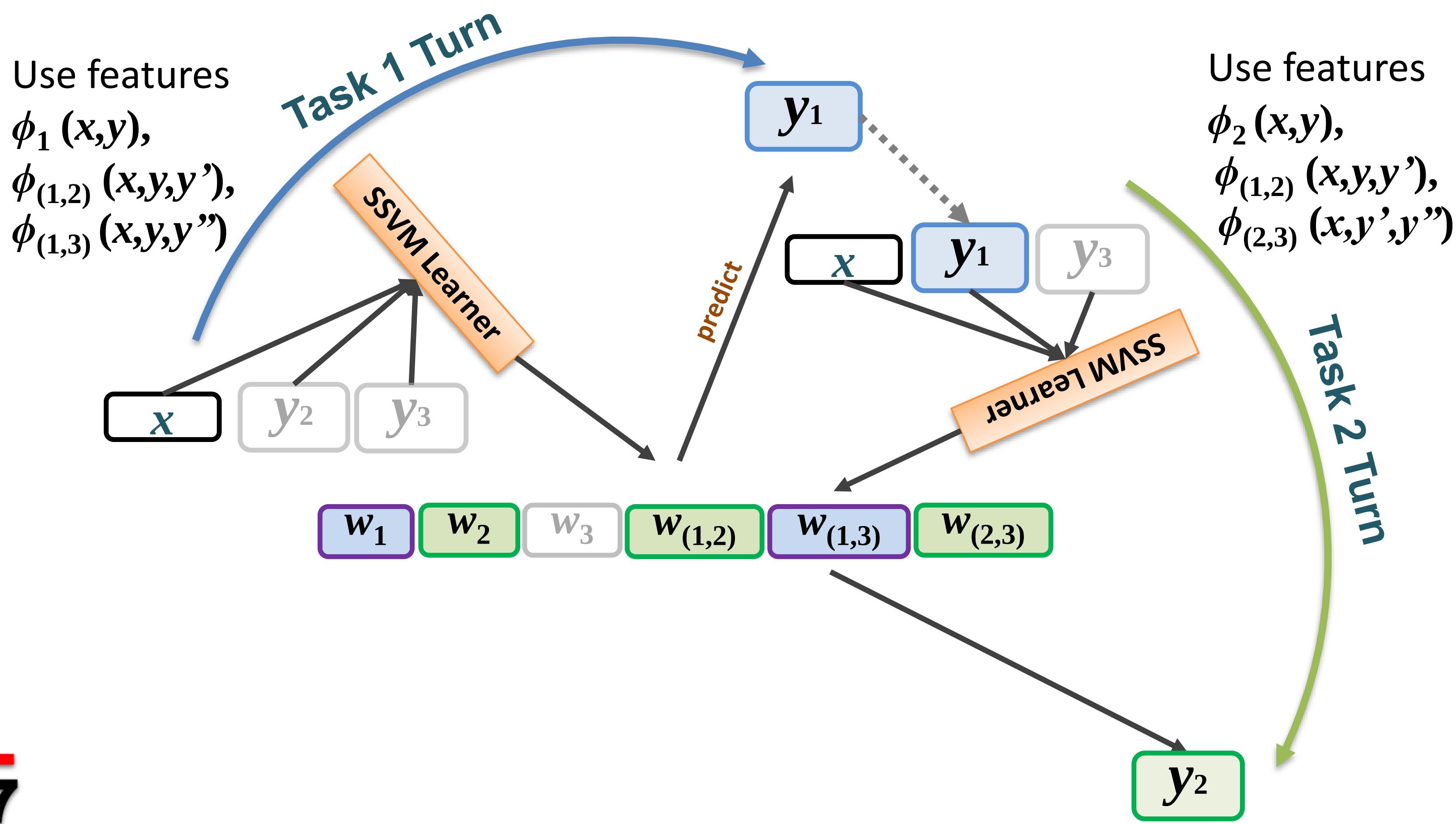
ACML 2017

Cyclic Architecture

Shared-Weight-Cyclic Training

Step 1: Define a order: Task 1 → Task 2 → Task 3

Step 2: Predict initial outputs: y_1 y_2 y_3

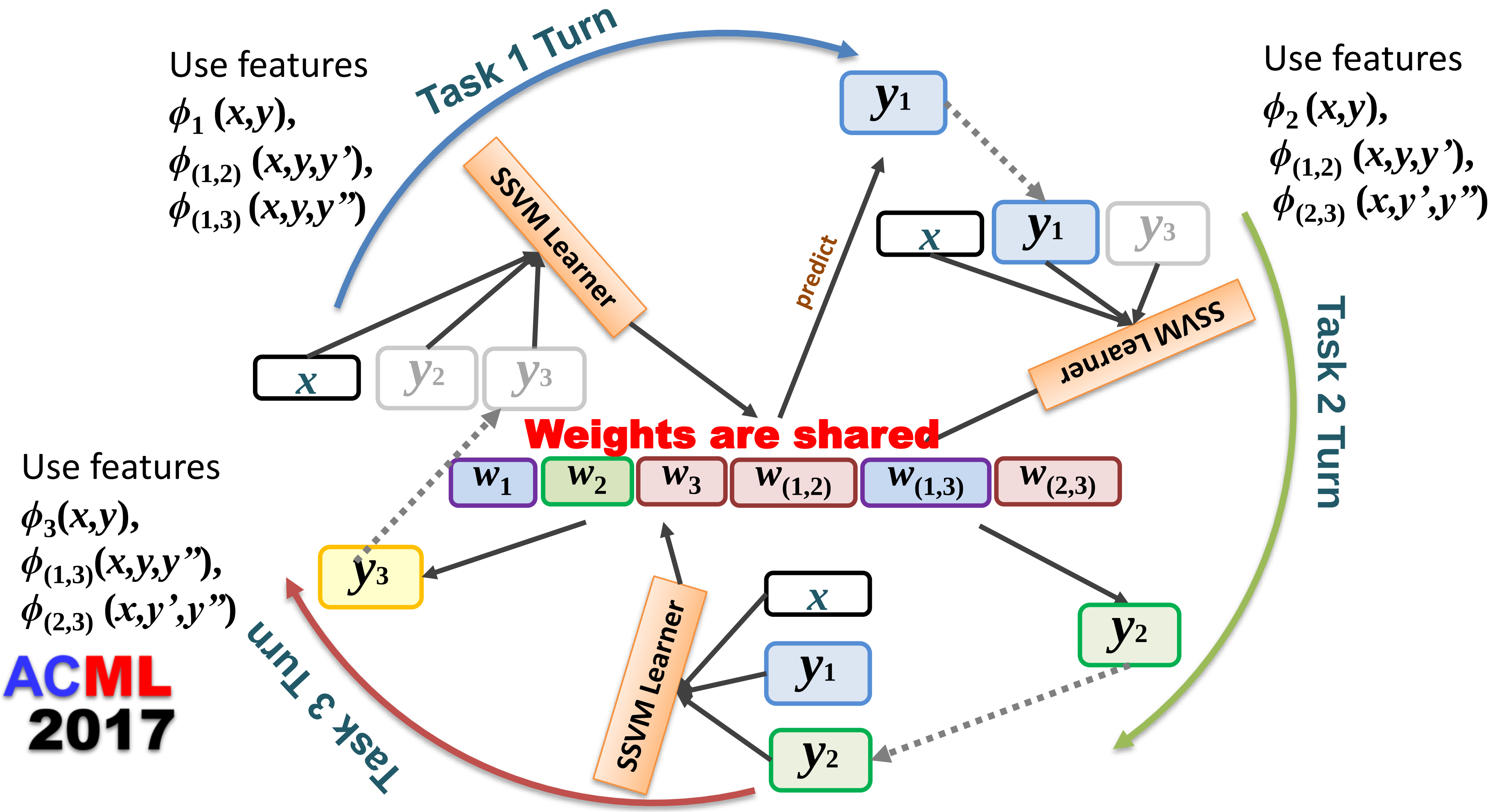


Cyclic Architecture

Shared-Weight-Cyclic Training

Step 1: Define a order: Task 1 → Task 2 → Task 3

Step 2: Predict initial outputs: y_1 y_2 y_3



ACML 2017

Experimental Setup

	ACE2005	TAC-KBP2015
Datasets:	**ACE2005** ACE-to-Wiki annotation Train/Dev/Test 338/144/117	**TAC-KBP2015** Train/Dev/Test 132/36/167
Knowledge Base:	**Wikipedia** (2015 dump)	**Freebase** (2014 dump)
Evaluation:	*Coref.* → CoNLL (MUC, BCube, CEAF*e* → average → CoNLL) *NER* → Hamming *Linking* → Hamming	*Within.Coref* → CoNLL *Cross.Coref* → CEAF*m* *NER & Linking* → NERLC

Combined accuracy of NER and Linking

ACML 2017

All metrics are accuracies (larger is better)

Results

Joint Architecture Performance

ACE05 Test Set Performance

Algms.	Coreference				NER	Link	Train time
	MUC	*BCube*	*CEAFe*	*CoNLL*	*Accu.*	*Accu.*	
Berkeley	81.41	74.7	72.93	76.35	85.6	76.78	31min
a. Results of Joint Architecture without Pruning							
STSP	80.28	73.26	71.58	75.04	82.24	75.36	9min
Joint w. Rand Init	80.23	73.79	72.03	75.35	82.20	76.99	48min
Joint w. Good init	82.18	76.57	74.00	77.58	85.71	78.77	34min

TAC15 Test Set Performance

Algm.	NER	Link	NERLC	Within. Coref	Cross. Coref	Train.
	Accu.	*Accu.*	*Accu.*	*CoNLL*	*CEAFm*	*time*
Rank-1st	87	-	73.7	-	80	-
Berkeley	88.9	74.8	72.8	82.98	80.8	6m29s
a. Results of Joint Architecture without Pruning						
STSP	87.3	76.2	70.9	81.21	78.8	2m41s
Joint w. Rand. Ini	87.1	71.17	68.33	81.31	78.4	7m19s
Joint w. Good. Ini	89.72	76.98	74.43	82.8	81.3	6m11s

1. **Joint-Good-Init > STSP**
   Interdependency, captured by inter-task features, does benefit the system.
2. **Joint-Good-Init > Joint-Rand-Init**
   Search-based inference for large structured prediction problems suffers from local optima and is mitigated by a good initialization.
3. Search-based MTSP is competitive or better than the state-of-the-art system.

Results

Joint Architecture Performance

OSU Oregon State UNIVERSITY

ACE05 Test Set Performance

Algms.	Coreference				NER	Link	Train time
	MUC	*BCube*	*CEAFe*	*CoNLL*	*Accu.*	*Accu.*	
Berkeley	81.41	74.7	72.93	76.35	85.6	76.78	31min
a. Results of Joint Architecture without Pruning							
STSP	80.28	73.26	71.58	75.04	82.24	75.36	9min
Joint w. Rand Init	80.23	73.79	72.03	75.35	82.20	76.99	48min
Joint w. Good init	82.18	76.57	74.00	77.58	85.71	78.77	34min
b. Results of Joint Architecture with Pruning							
Score-agnostic	81.10	75.79	74.33	77.07	85.63	78.71	16min
Score-sensitive	82.81	75.77	74.96	**77.85**	**87.18**	**80.28**	37min

TAC15 Test Set Performance

Algm.	NER	Link	NERLC	Within. Coref	Cross. Coref	Train.
	Accu.	*Accu.*	*Accu.*	*CoNLL*	*CEAFm*	*time*
Rank-1st	87	-	73.7	-	80	-
Berkeley	88.9	74.8	72.8	82.98	80.8	6m29s
a. Results of Joint Architecture without Pruning						
STSP	87.3	76.2	70.9	81.21	78.8	2m41s
Joint w. Rand. Ini	87.1	71.17	68.33	81.31	78.4	7m19s
Joint w. Good. Ini	89.72	76.98	74.43	82.8	81.3	6m11s
b. Results of Joint Architecture with Pruning						
Score-agnostic	89.54	76.84	74.31	82.99	81.4	4m15s
Score-sensitive	89.33	77.68	**74.63**	**83.17**	**81.3**	9m2s

1. **Joint-Good-Init > STSP**
   Interdependency, captured by inter-task features, does benefit the system.
2. **Joint-Good-Init > Joint-Rand-Init**
   Search-based inference for large structured prediction problems suffers from local optima and is mitigated by a good initialization.
3. Search-based MTSP is competitive or better than the state-of-the-art system.
4. Score-sensitive pruning of joint MTSP performs the best and takes most time

ACML 2017

Results

Cyclic Architecture Performance

ACE05 Test Set Performance

Algms.	Coreference MUC	Coreference BCube	Coreference CEAFe	Coreference CoNLL	NER Accu.	Link Accu.	Train time
Berkeley	81.41	74.7	72.93	76.35	85.6	76.78	31min
a. Results of Joint Architecture without Pruning							
STSP	80.28	73.26	71.58	75.04	82.24	75.36	9min
Joint w. Rand Init	80.23	73.79	72.03	75.35	82.20	76.99	48min
Joint w. Good init	82.18	76.57	74.00	77.58	85.71	78.77	34min
b. Results of Joint Architecture with Pruning							
Score-agnostic	81.10	75.79	74.33	77.07	85.63	78.71	16min
Score-sensitive	82.81	75.77	74.96	77.85	87.18	80.28	37min
c. Results of Cyclic Architecture							
Unshard-Wt-Cyclic	81.83	76.05	73.99	77.29	84.18	80.67	**11min**
Shared-Wt-Cyclic	80.97	75.22	73.39	76.53	82.16	79.60	**10min**

TAC15 Test Set Performance

Algm.	NER Accu.	Link Accu.	NERLC Accu.	Within. Coref CoNLL	Cross. Coref CEAFm	Train. time
Rank-1st	87	-	73.7	-	80	-
Berkeley	88.9	74.8	72.8	82.98	80.8	6m29s
a. Results of Joint Architecture without Pruning						
STSP	87.3	76.2	70.9	81.21	78.8	2m41s
Joint w. Rand. Ini	87.1	71.17	68.33	81.31	78.4	7m19s
Joint w. Good. Ini	89.72	76.98	74.43	82.8	81.3	6m11s
b. Results of Joint Architecture with Pruning						
Score-agnostic	89.54	76.84	74.31	82.99	81.4	4m15s
Score-sensitive	89.33	77.68	74.63	83.17	81.3	9m2s
c. Results of Cyclic Architecture						
Unshared-Wt-Cyc	89.57	77.68	74.6	82.08	80.5	**3m52s**
Shared-Wt-Cyc	87.95	73.65	71.32	80.54	77.9	**2m56s**

- Competitive accuracy, and much faster training
- Unshared weights perform better than shared weights

Summary

1. Search-based multi-task structured prediction outperforms prior work based on graphical models on all 3 entity analysis tasks.

2. Studied three learning and inference architectures: *pipeline, cyclic, and joint*, with trade-offs between accuracy and speed.

3. The joint architecture with score-sensitive pruning performs the best.

4. The cyclic architecture with unshared weights is competitive in accuracy and faster to train.

Thank You!