New Directions in Search-Based Structured Prediction: Multi-Task Learning and Integration of Deep Models

PhD Final Exam Presentation

Committee:

Prasad Tadepalli, Janardhan Rao Doppa, Xiaoli Fern, Weng-Keen Wong

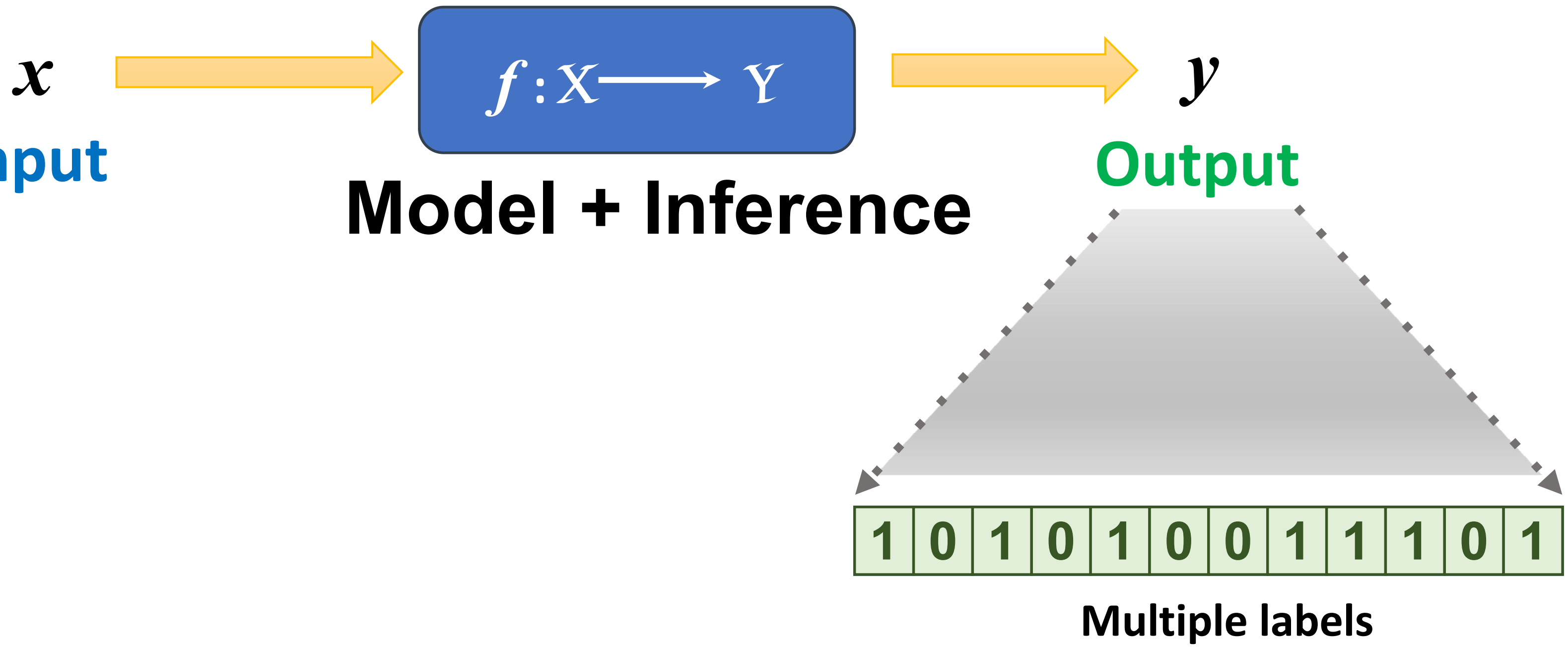
GCR: Leonard Coop

Chao Ma

July 15th, 2019

Introduction

Structured Prediction:



Introduction

Structured Prediction:

x → $f : X \longrightarrow Y$ → y

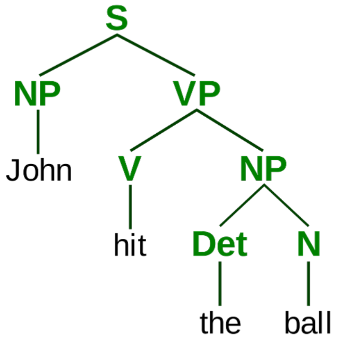
Input

Model + Inference

Output

Syntactic Parsing

John hit the ball →



Introduction

Structured Prediction:

x → $f: X \longrightarrow Y$ → y

Input **Model + Inference** **Output**

Syntactic Parsing

John hit the ball →

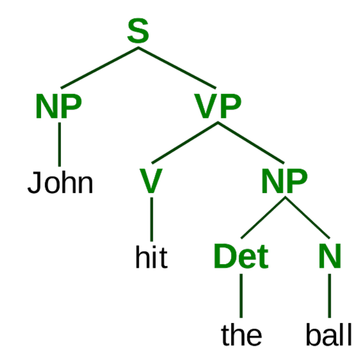
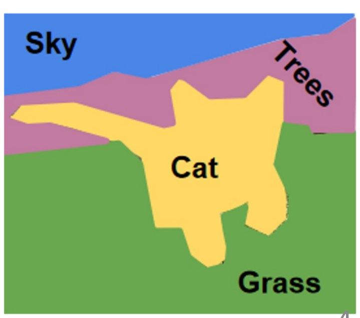


Image Scene Labeling

→



Introduction

Structured Prediction:

x → $f: X \longrightarrow Y$ → y

Input | Output

Model

$$f(x, y)$$

e.g.

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

NN: $$f(x, y) = DNN\,\underbrace{(x, y)}_{\text{Representation Vector}}$$

Inference

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

Introduction

Search-Based ***Structured Prediction:***

x → $f: X \longrightarrow Y$ → y

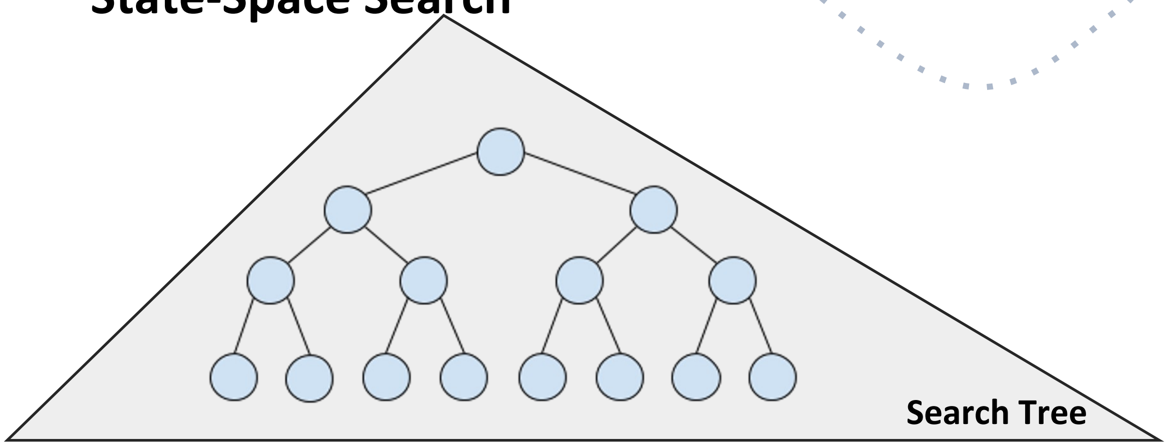
Input **Output**

Inference *Model*

$$\hat{y} = \underset{y}{argmax} \; f(x, y)$$

- Search Space Formulation
- Search Algorithm
- Scoring function to guide the search

State-Space Search



Structured Prediction Frameworks

We can classify most of the structured prediction approaches into this four cell table.

	Non-Deep Models	Deep Models
Non-Search-Based	CRFs, SSVM, SPerceptron	DSM, SPEN, DVN
Search-Based	LaSO, **PruneScore**, HC-Search, **MTSP**	Seq2Seq-BSO **HC-Nets**

This Thesis

Partial vs. Complete Output Space

State Space: *Partial* vs. *Complete* Output Space

Input: congratulations

Outputs:	Outputs:
$y =$????????????	$y =$ cangrotulerians
↓	↓
$y =$ c???????????	$y =$ congrotulerians
↓	↓
$y =$ co??????????	$y =$ congrotulerions
↓	↓
$y =$ con?????????	$y =$ congratulerions
⋮	⋮

Contributions

1. Developed ***Prune-and-Score*** to improve the accuracy of greedy policy based structured prediction with large action spaces.

1. Studied three learning architectures for multi-task structured prediction (**MTSP**) problems with different trade-offs between speed and accuracy.

1. Proposed a ***HC*-Nets** framework that synergistically combines the advantages of output space search based structured prediction methods and deep models.

Prune-and-Score: Learning Greedy Policy for Structured Prediction

For some problem, even with greedy search, the branching factor is still too large to make the correct decision.

This work tries to answer following questions:

- **Can we prune the action space at each step to reduce the branching factor and improve the accuracy?**
- **Can we formulate the imitation learning problem to an offline rank learning problem?**

Coreference Resolution: The Problem

Extracted Mentions

*"[**Barack Obama**] nominated [**Hillary Clinton**] as his [**secretary of state**] on Monday. [**He**] chose [**her**] because [**she**] had foreign affair experience as a former [**First Lady**]."*

Input: x

↓

Coreference Resolver

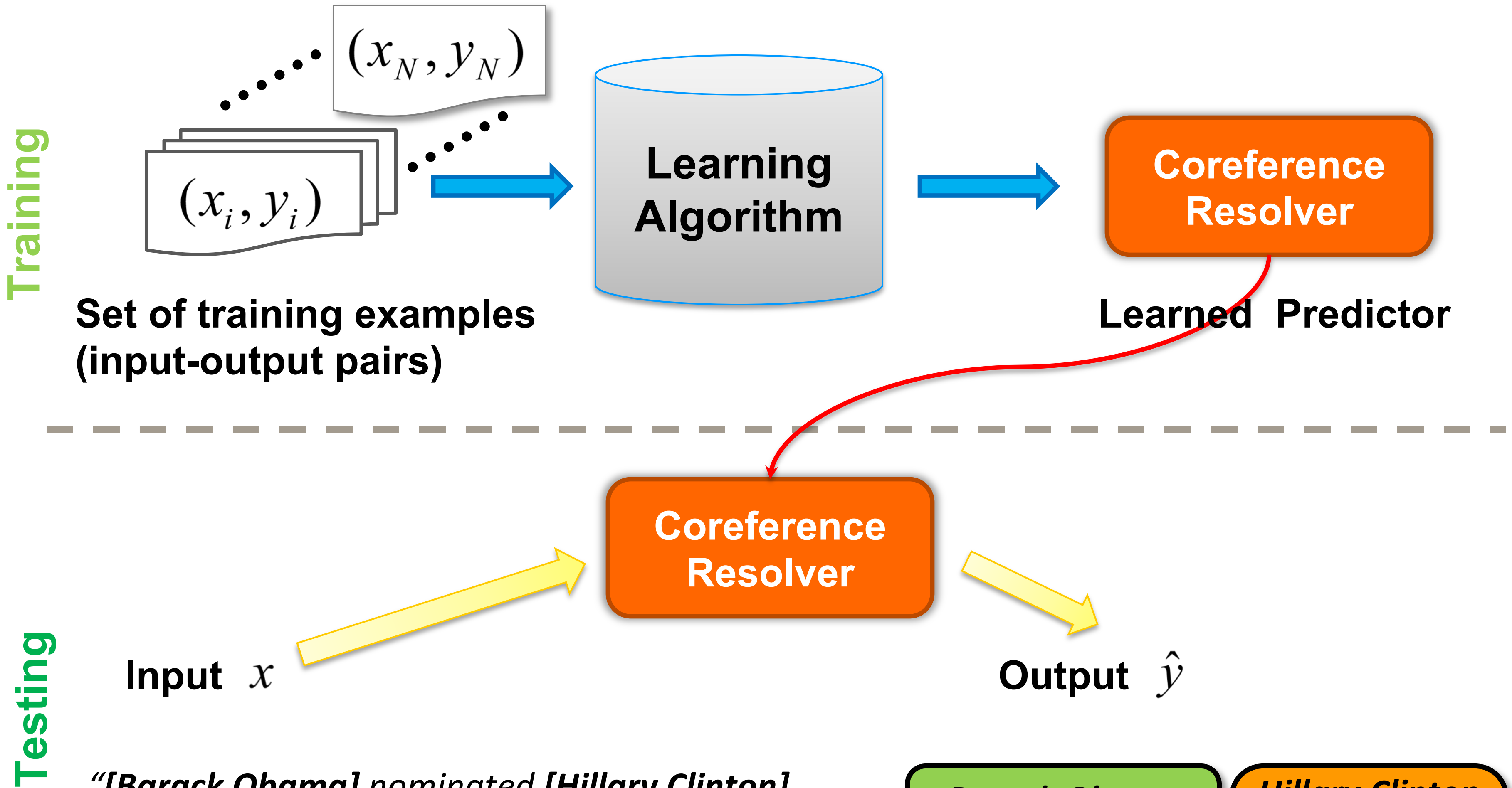
↓

Output: y

Coreference Clustering

Barack Obama *He*	*Hillary Clinton* *First Lady* *Her* *She*
Secretary of state	

Coreference Resolution: Problem Setup



"[Barack Obama] nominated [Hillary Clinton] as his [secretary of state] on Monday. [He] chose [her] because [she] had foreign affair experience as a former [First Lady]."

Barack Obama
He

Secretary of state

Hillary Clinton
First Lady
Her
She

Coreference Resolution: Greedy Search

[Ramallah *(**[West Bank]**)**]**10-15* ***([AFP]*** *) –* ***[Eyewitnesses]*** *reported that* ***[Palestinians]*** *demonstrated today Sunday in the* ***[West Bank]*** *against the* ***[Sharm el-Sheikh]*** *summit to be held in* ***[Egypt]*** *tomorrow Monday. In* ***[Ramallah]****,* ***[around 500 people]*** *took to* ***[the town]****'s streets chanting slogans denouncing the summit ...*

S_0

Coreference Resolution: Greedy Search

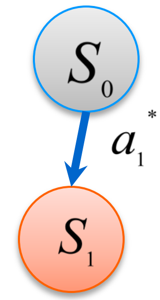
[Ramallah ([West Bank])] 10-15 ([AFP]) – [Eyewitnesses] reported that [Palestinians] demonstrated today Sunday in the [West Bank] against the [Sharm el-Sheikh] summit to be held in [Egypt] tomorrow Monday. In [Ramallah], [around 500 people] took to [the town]'s streets chanting slogans denouncing the summit ...

Current mention: ***[Ramallah (West Bank)]***

(Partial) Co-reference result:

Coreference Resolution: Greedy Search

[Ramallah ([West Bank])] 10-15 ([AFP]) – [Eyewitnesses] reported that [Palestinians] demonstrated today Sunday in the [West Bank] against the [Sharm el-Sheikh] summit to be held in [Egypt] tomorrow Monday. In [Ramallah], [around 500 people] took to [the town]'s streets chanting slogans denouncing the summit ...



Current mention: ***[Ramallah (West Bank)]***

(Partial) Co-reference result:

Ramallah (West Bank)

Coreference Resolution: Greedy Search

[Ramallah** ([**West Bank**])**]10-15 ***([AFP]***) –
[Eyewitnesses] *reported that* ***[Palestinians]***
demonstrated today Sunday in the ***[West Bank]***
against the ***[Sharm el-Sheikh]*** *summit to be held in*
[Egypt] *tomorrow Monday. In* ***[Ramallah]***, ***[around 500 people]*** *took to* ***[the town]****'s streets chanting*
slogans denouncing the summit ...

Current mention: ***[Sharm el-Sheikh]***

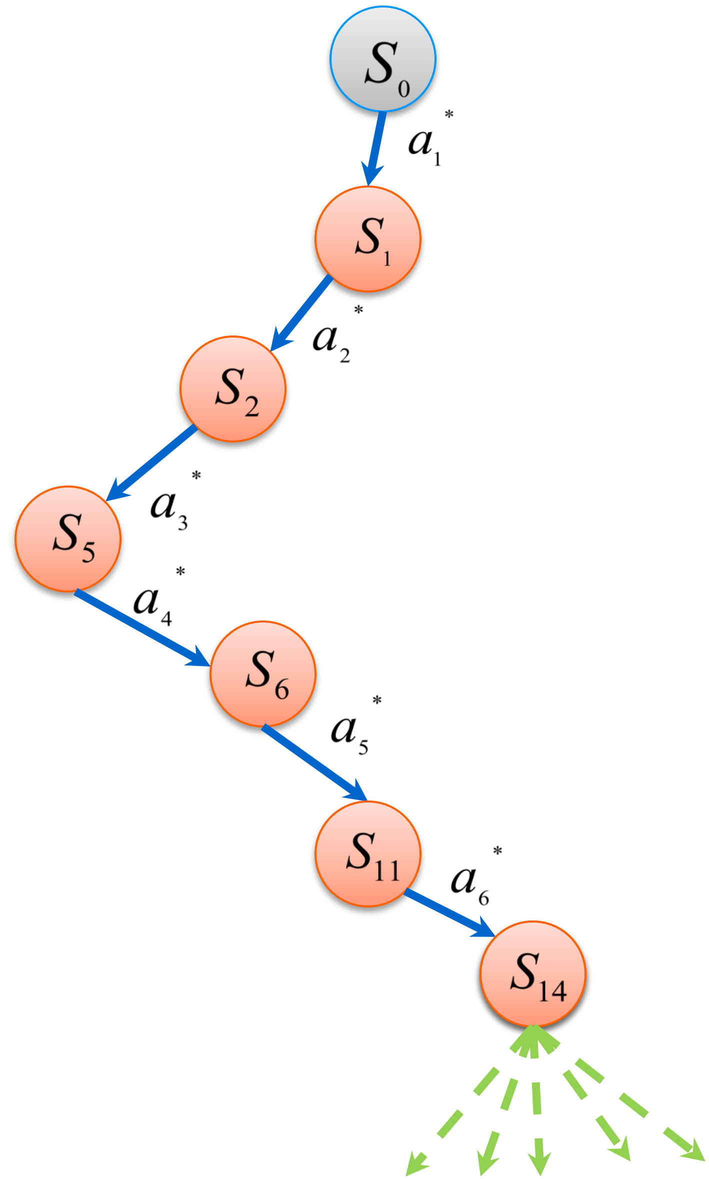
(Partial) Co-reference result:

***Ramallah** (**West Bank**)*

West Bank
West Bank

AFP
Eyewitnesses

Palestinians



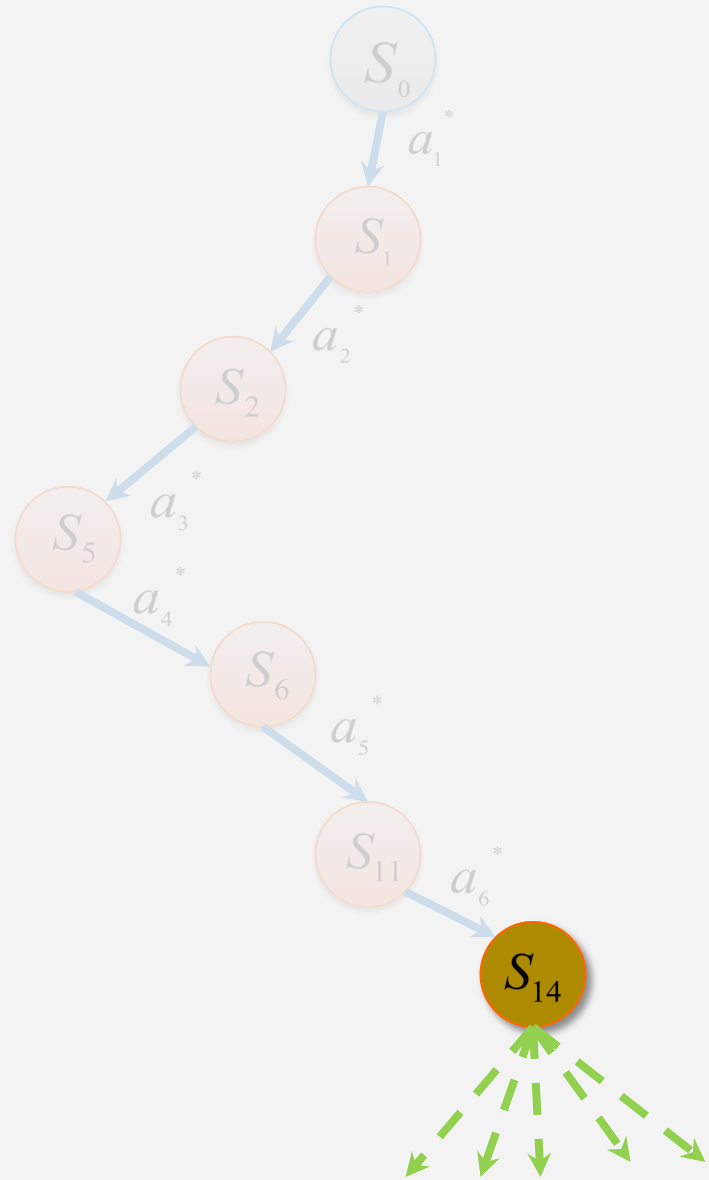
Coreference Resolution: Greedy Search

***[Ramallah** ([**West Bank**])**]**10-15 **([AFP]**) –*
[Eyewitnesses] *reported that* ***[Palestinians]*** *demonstrated today Sunday in the* ***[West Bank]*** *against the* ***[Sharm el-Sheikh]*** *summit to be held in* ***[Egypt]*** *tomorrow Monday. In* ***[Ramallah]***, ***[around 500 people]*** *took to* ***[the town]****'s streets chanting slogans denouncing the summit …*

Current mention: ***[Sharm el-Sheikh]***

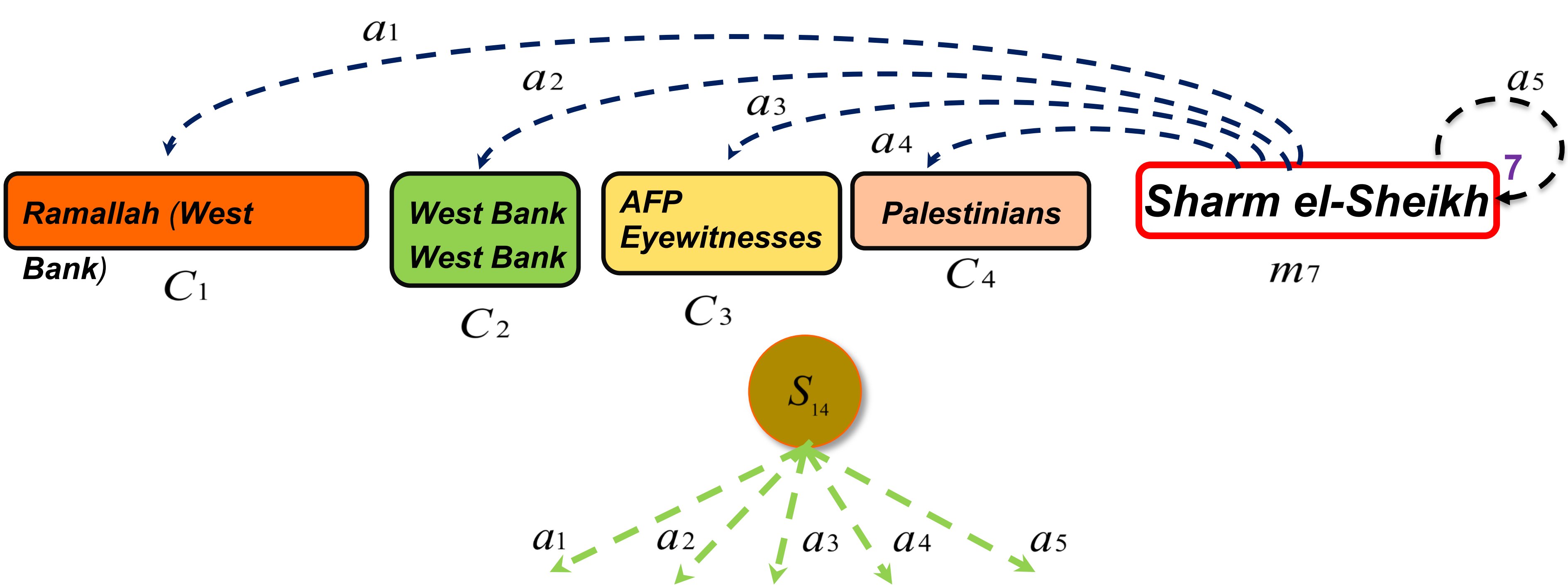
(Partial) Co-reference result:

- *Ramallah (West Bank)*
- *West Bank* / *West Bank*
- *AFP* / *Eyewitnesses*
- *Palestinians*



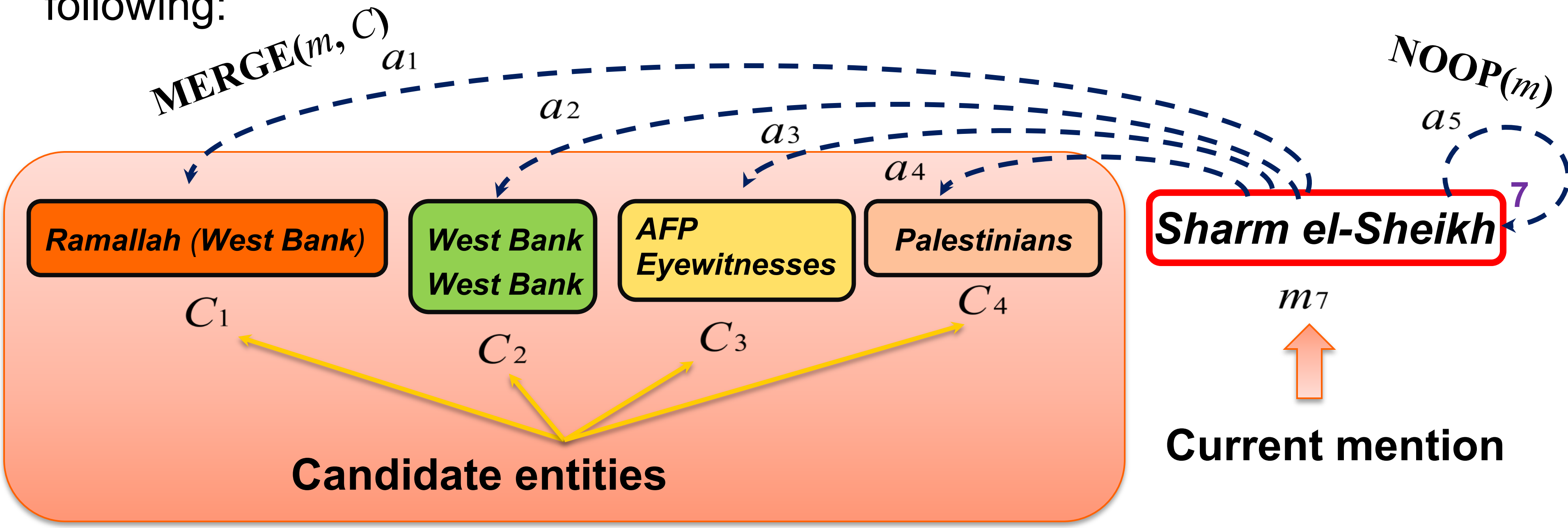
Successor Function

[Ramallah ([West Bank])] 10-15 ([AFP]) – [Eyewitnesses] reported that [Palestinians] demonstrated today Sunday in the [West Bank] against the [Sharm el-Sheikh] summit to be held in **[Egypt]** tomorrow Monday. In **[Ramallah]**, **[around 500 people]** took to **[the town]**'s streets chanting slogans denouncing the summit ...

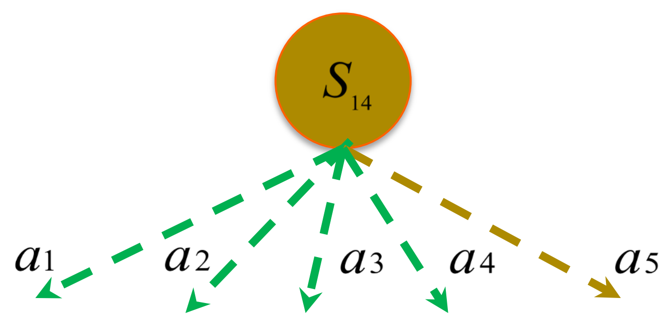


Successor Function

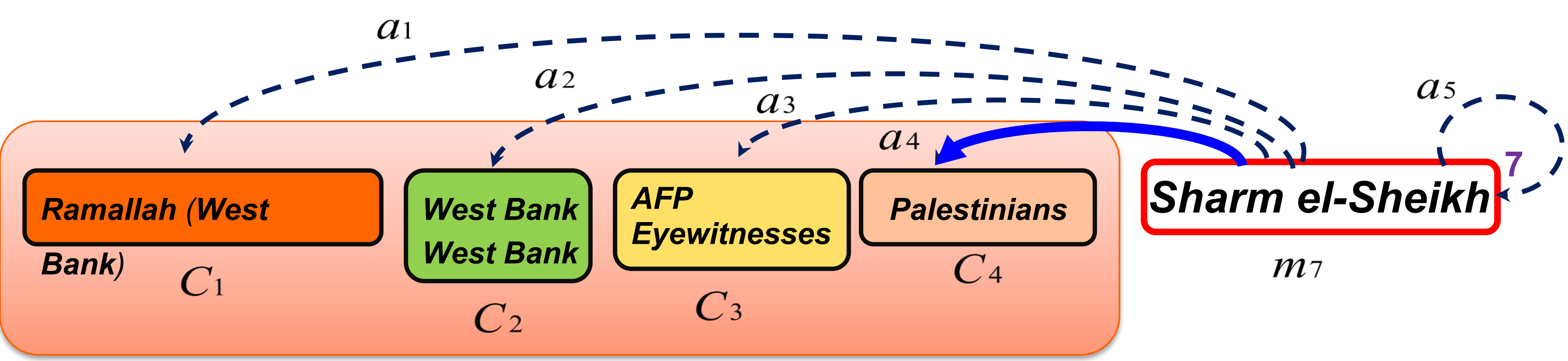
At any state at depth t, we will have several candidate actions as following:



State: partial coreference output



Prune-and-Score for Greedy Search: Illustrate



All candidate actions: $A(s) = \{a_1, a_2, a_3, a_4, a_5\}$

Pruner $F(prune)$ **with parameter** b

Pruning action by keeping top b (here $b = 2$).

$A'(s) = \{a_3, a_4\}$

Scorer $F(score)$

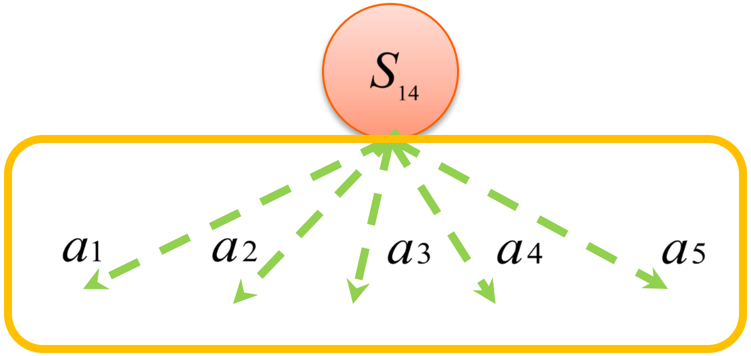
Picking the best action.

$\hat{a}$

Prune-and-Score for Greedy Search: Illustrate

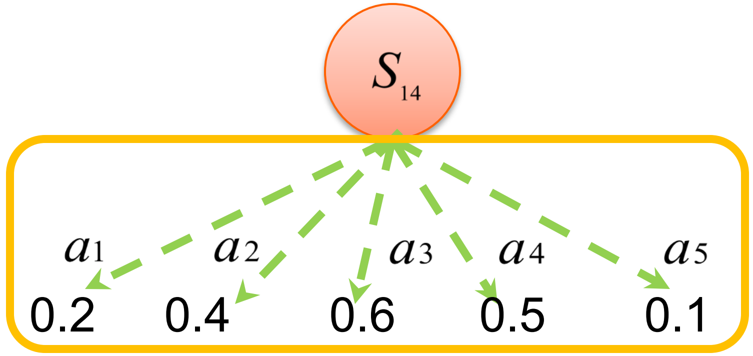
S_{14}

Prune-and-Score for Greedy Search: Illustrate



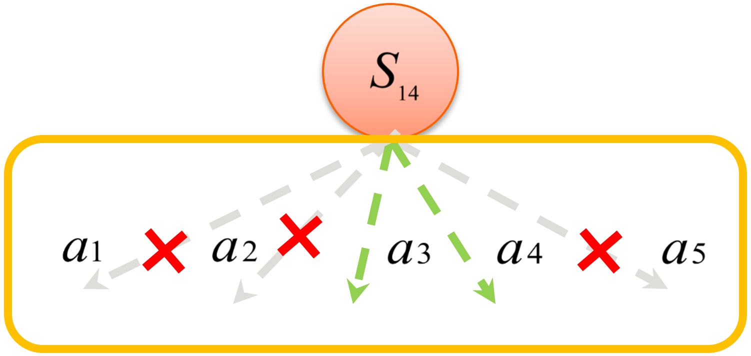
Pruning

Prune-and-Score for Greedy Search: Illustrate



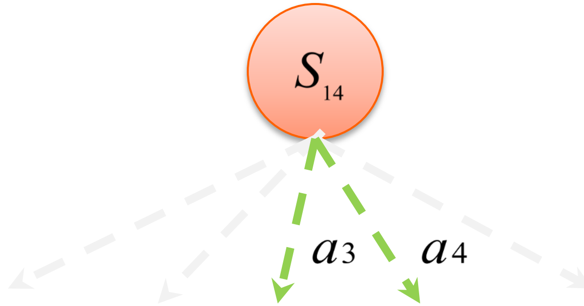
Pruning

Prune-and-Score for Greedy Search: Illustrate

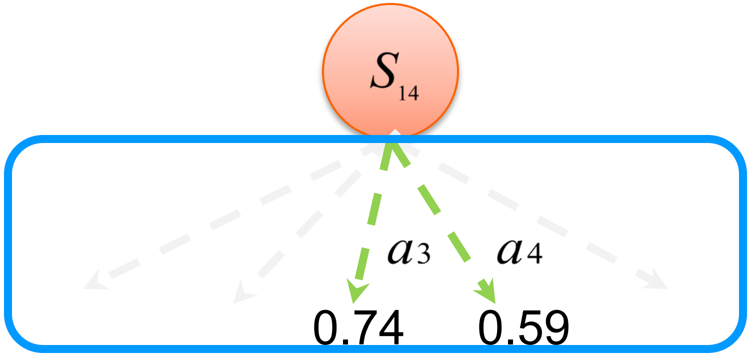


Pruning (b=2)

Prune-and-Score for Greedy Search: Illustrate

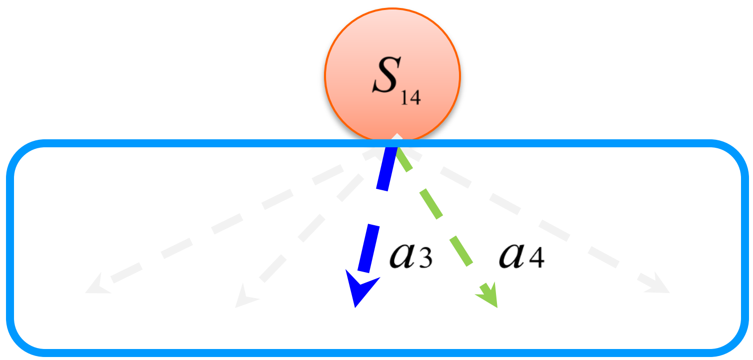


Prune-and-Score for Greedy Search: Illustrate



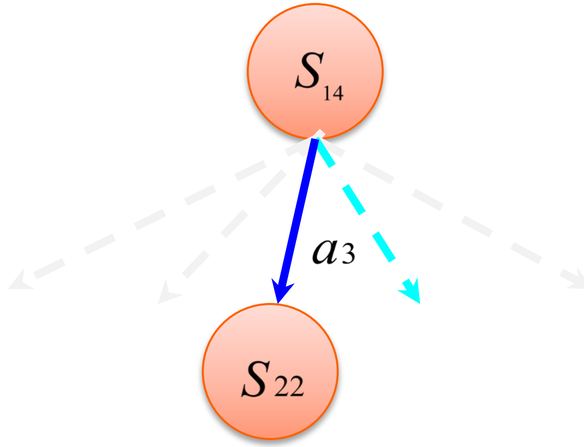
Scoring

Prune-and-Score for Greedy Search: Illustrate



Scoring

Prune-and-Score for Greedy Search: Illustrate



Prune-and-Score: Learning Approach

We optimize the overall loss of the Prune-and-Score approach in a ***stage-wise*** manner.

Stage 1: Learn pruning function to optimize the pruning error.

$$\hat{\mathcal{F}}_{prune} \approx \arg\min_{\mathcal{F}_{prune} \in \mathbf{F_p}} \quad \epsilon_{prune}$$

conditioned on

Stage 2: Learn scoring function conditioned on the learned pruning function.

$$\hat{\mathcal{F}}_{score} \approx \arg\min_{\mathcal{F}_{score} \in \mathbf{F_s}} \quad \epsilon_{score|\hat{\mathcal{F}}_{prune}}$$

Experimental Setup

Datasets

MUC 6: Message Understanding Conference (MUC6, 1995)
Train/Dev/Test: 268/68/107 documents

ACE 2004: Automatic Content Extraction (NIST, 2004)
Train/Dev/Test: 195/30/30 documents

Evaluation Metrics

MUC F1, **BCubed F1**, **CEAF F1**.

Base Ranker Learner

LambdaMART (Burges, 2010), implemented in **RankLib**

Baseline Approaches

Only Scoring Function
UIUC: $\mathbf{CPL^3M}$ (Chang et al., 2013)
JHU: **Easyfirst** (Stoyanov and Eisner, 2012)
Stanford: **Multi-Sieves** (Raghunathan et al., 2010)

Prune-and-Score vs. State-of-the-Art

Oregon State University

	ACE 2004			MUC 6		
F-1 score	**MUC**	**B-Cubed**	**CEAF**	**MUC**	**B-Cubed**	**CEAF**
Prune-Score	78.6	**83.04**	**79.42**	85.27	**80.49**	**67.83**
Only Scoring	75.4	80.75	78.58	83.76	77.11	64.91
JHU	**80.1**	81.8	-	**88.2**	77.5	-
UIUC	78.29	82.2	79.26	-	-	-

Summary of Prune-and-Score

- Coreference Resolution as a greedy search process
- **Key Idea:** Scoring Function $\longrightarrow$ Pruning Function + Scoring Function
- Apply the offline rank learner for imitation learning
- Achieved results that are comparable than the state-of-the-art

Multi-Task Structured Prediction (MTSP) for Entity Analysis

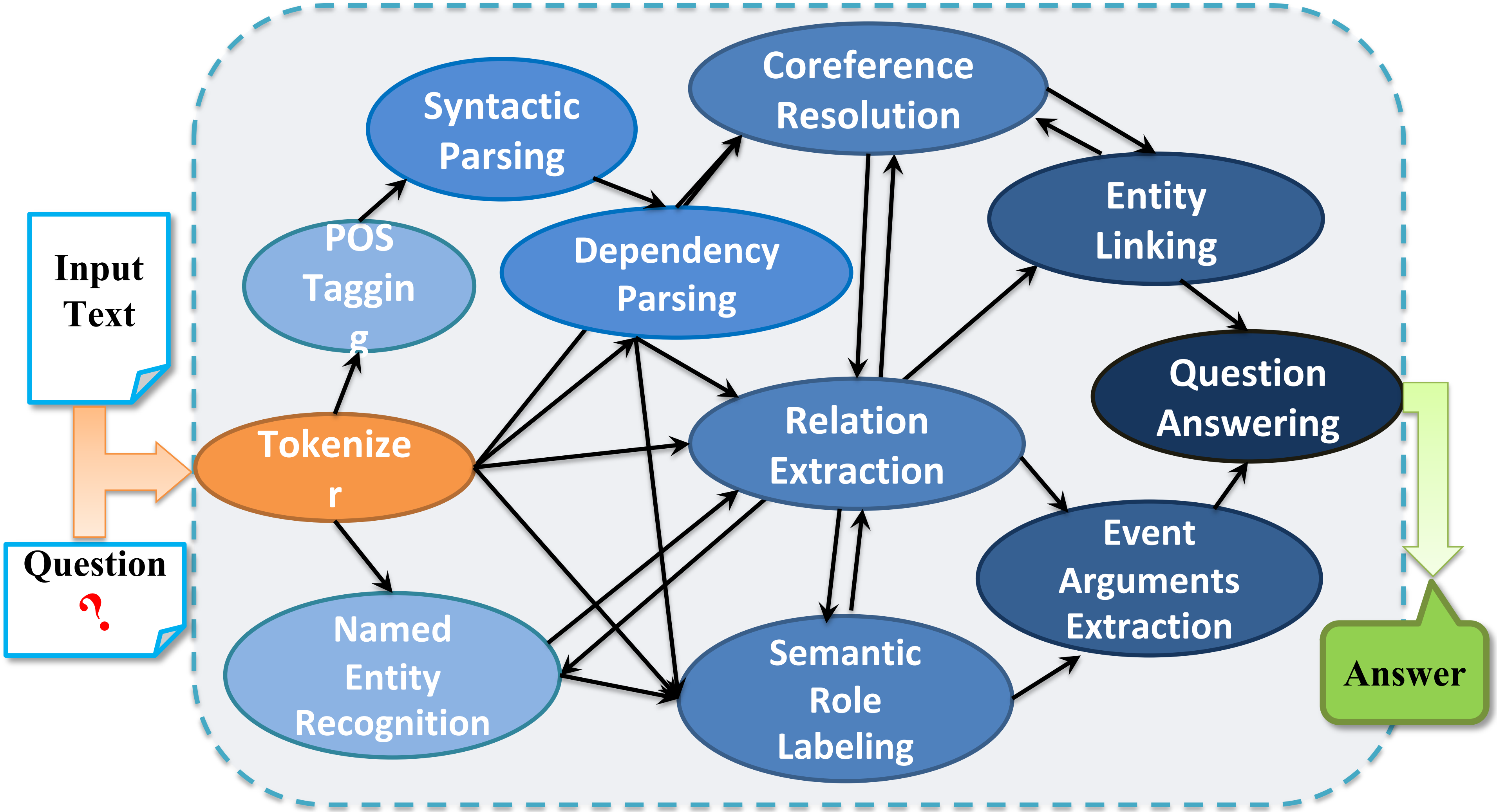
What if you want to solve multiple corelated structured prediction tasks with complete output space search?

We can concatenate the output of multiple tasks to form a super-output. But search on such super-output would be slow due to the huge branching factor.

We want to use complete output space so that we can extract high-order features to exploit the interdependencies between tasks

- **Can we do complete output space search for multiple tasks accurately and efficiently?**

Example of NLP Pipelines



A composite NLP system for Text Comprehension and Question Answering

Entity Analysis Tasks

The NLP tasks that are related to "***entity mentions***"

- **Named Entity Recognition**

Automatically find names of people, places, and organizations in text across many languages.

- **Coreference Resolution**

"I voted for Nader because he was most aligned with my values," she said.

- **Entity Linking**

"Paris is the capital of France"

wikipedia.org/wiki/Paris

wikipedia.org/wiki/France

Problem Setup

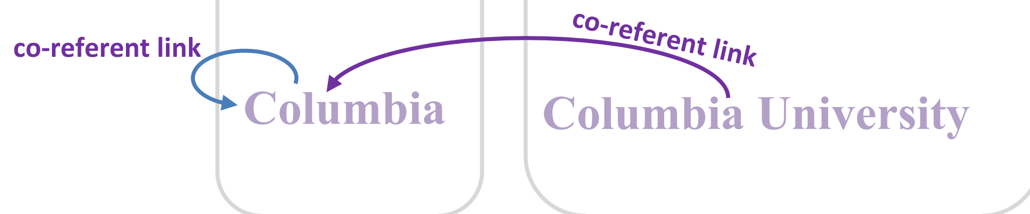
$\boldsymbol{i = 1}$ $\boldsymbol{i = 2}$

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

Problem Setup

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

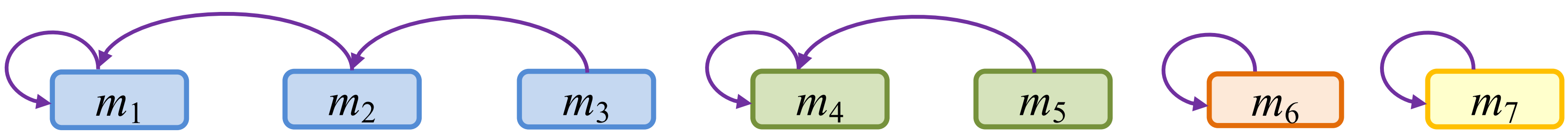
He left [**Columbia**] in 1983 with a BA degree, ... after graduating from [**Columbia University**], he worked as a community organizer in Chicago…



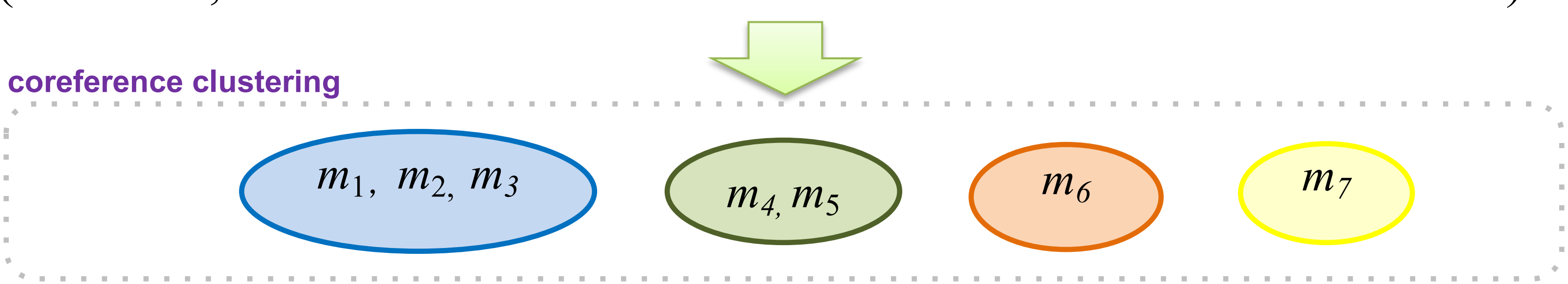
Coreference:

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

Left-linking Tree formulation for coreference resolution:



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



Problem Setup

$i = 1$ $i = 2$

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

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

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

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

Problem Setup

$i = 1$ $i = 2$

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

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

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

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

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

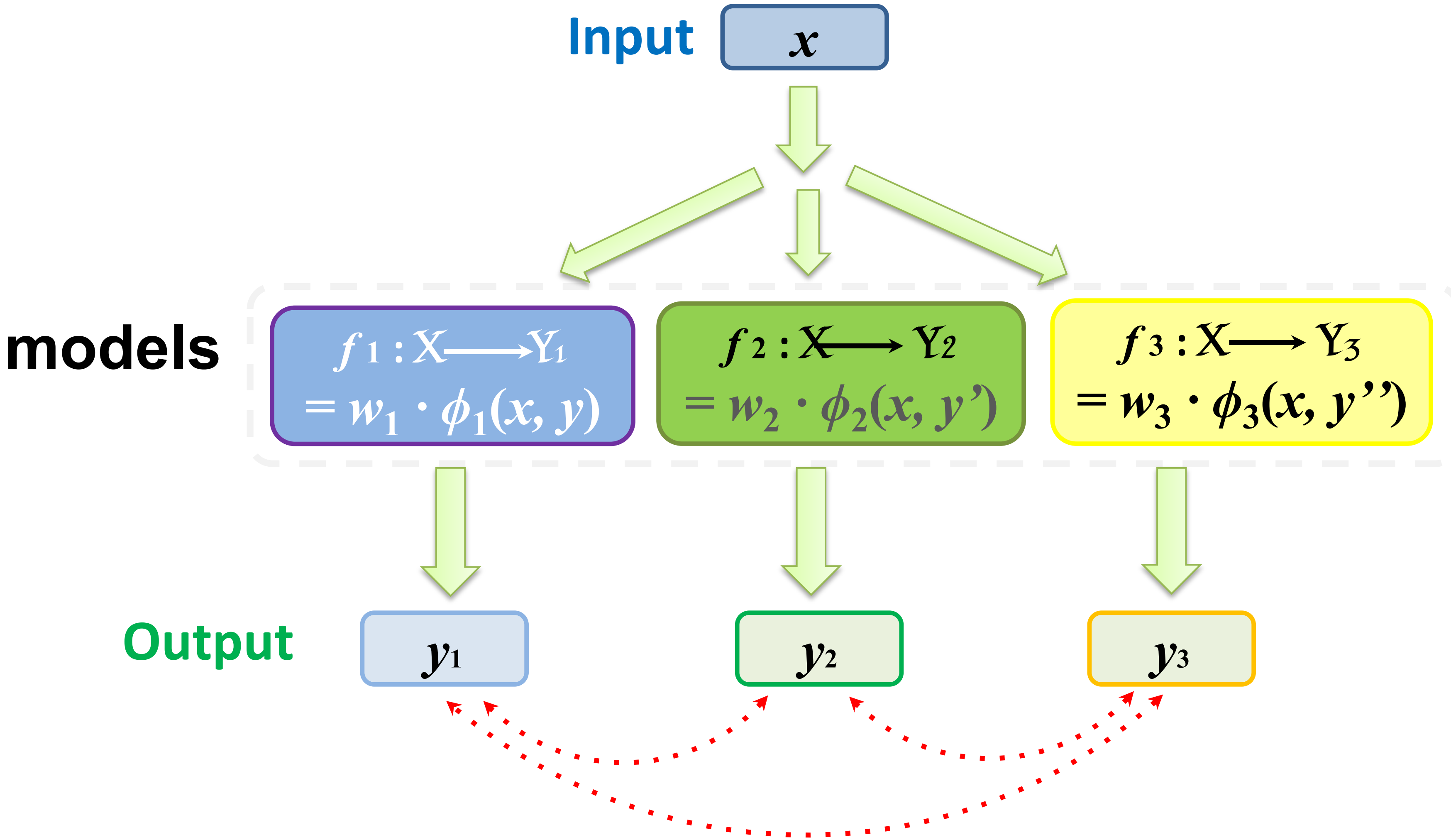
$y_i = \{$

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,
…

$\}$

Multi-Task Structure Prediction

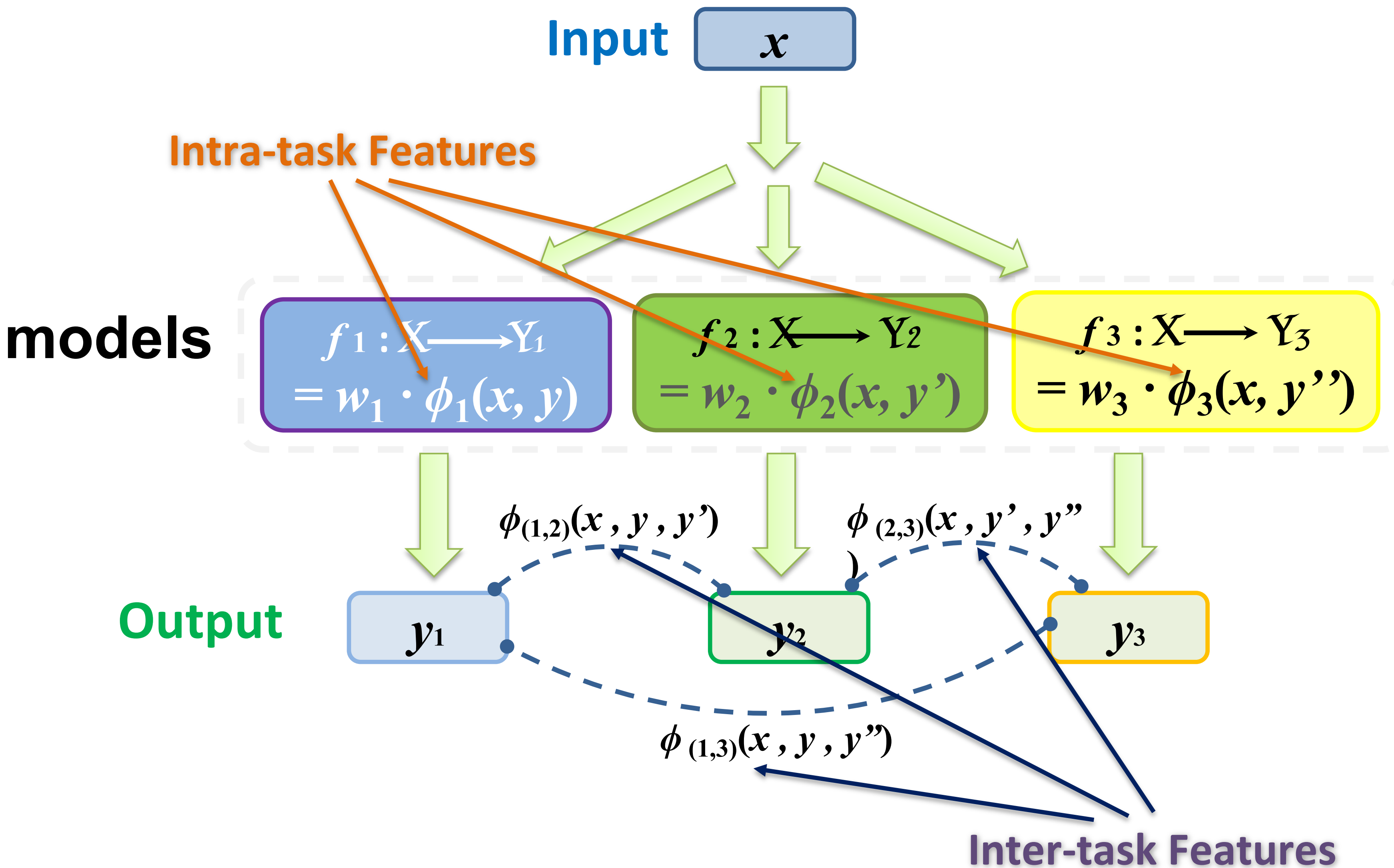
Multi-Task Structured Prediction (MTSP):



- ***How to exploit the interdependencies between tasks?***

Multi-Task Structure Prediction

Introduce Inter-task Features:



Inter-task Features

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 **Columbia University**

Coref-NER:

e.g.: Agreement of NER tags of two coreferent mentions

ORG == ORG

Coref-Link:

e.g.: Relation of KB entries of two coreferent mentions

University is-same-category University
Mathematics is-sub-category Mathematics education

NER-Link:

e.g.: NER-tag and Category pair indicator

Bonus to the co-related pair

(ORG, University)
(ORG, Institute)
(PER, President) …

Pipeline Architecture

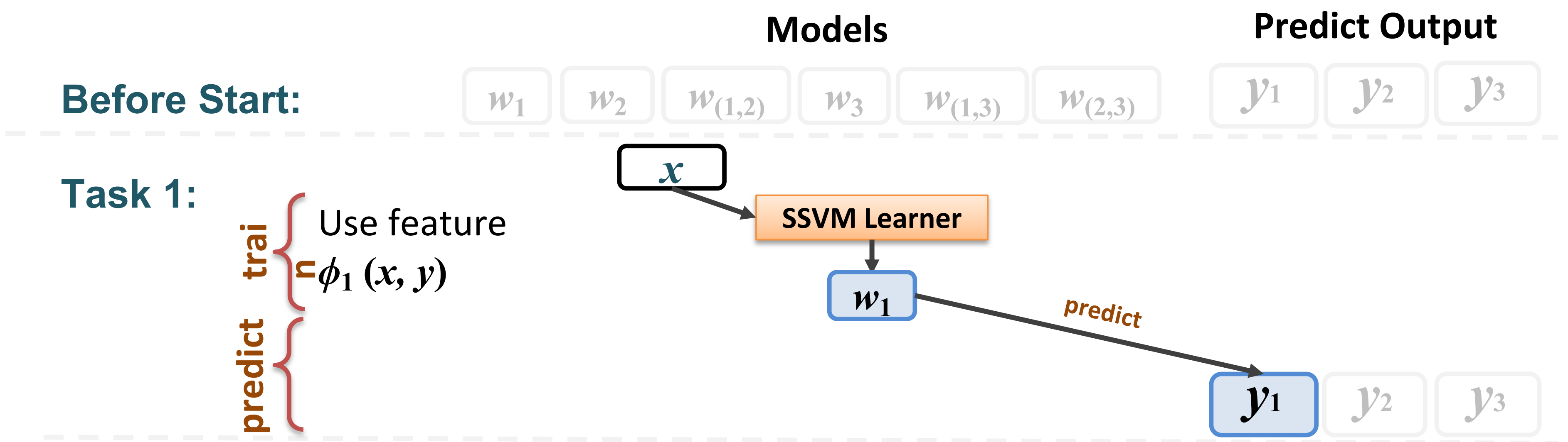
Learning k (= 3) independent models, one after another;

	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

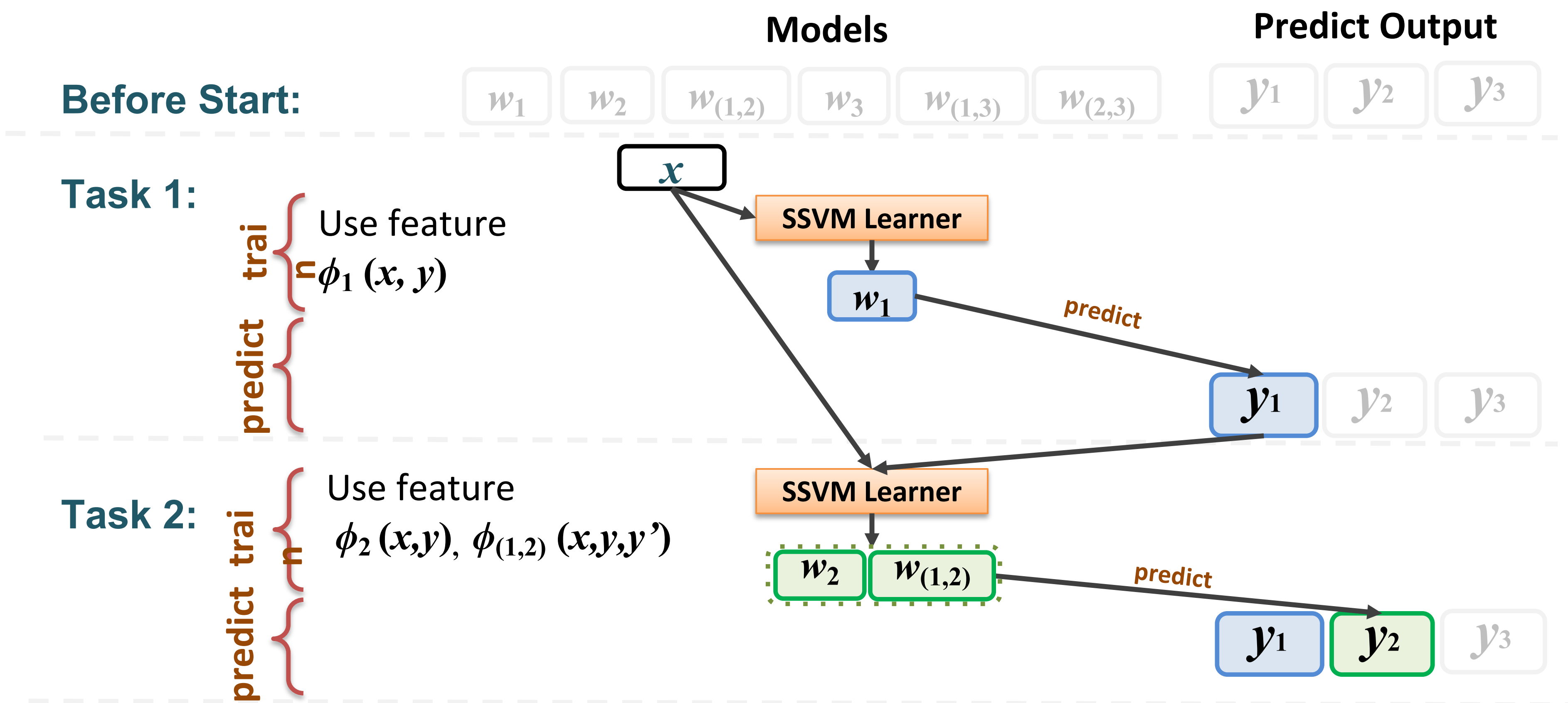
Pipeline Architecture

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



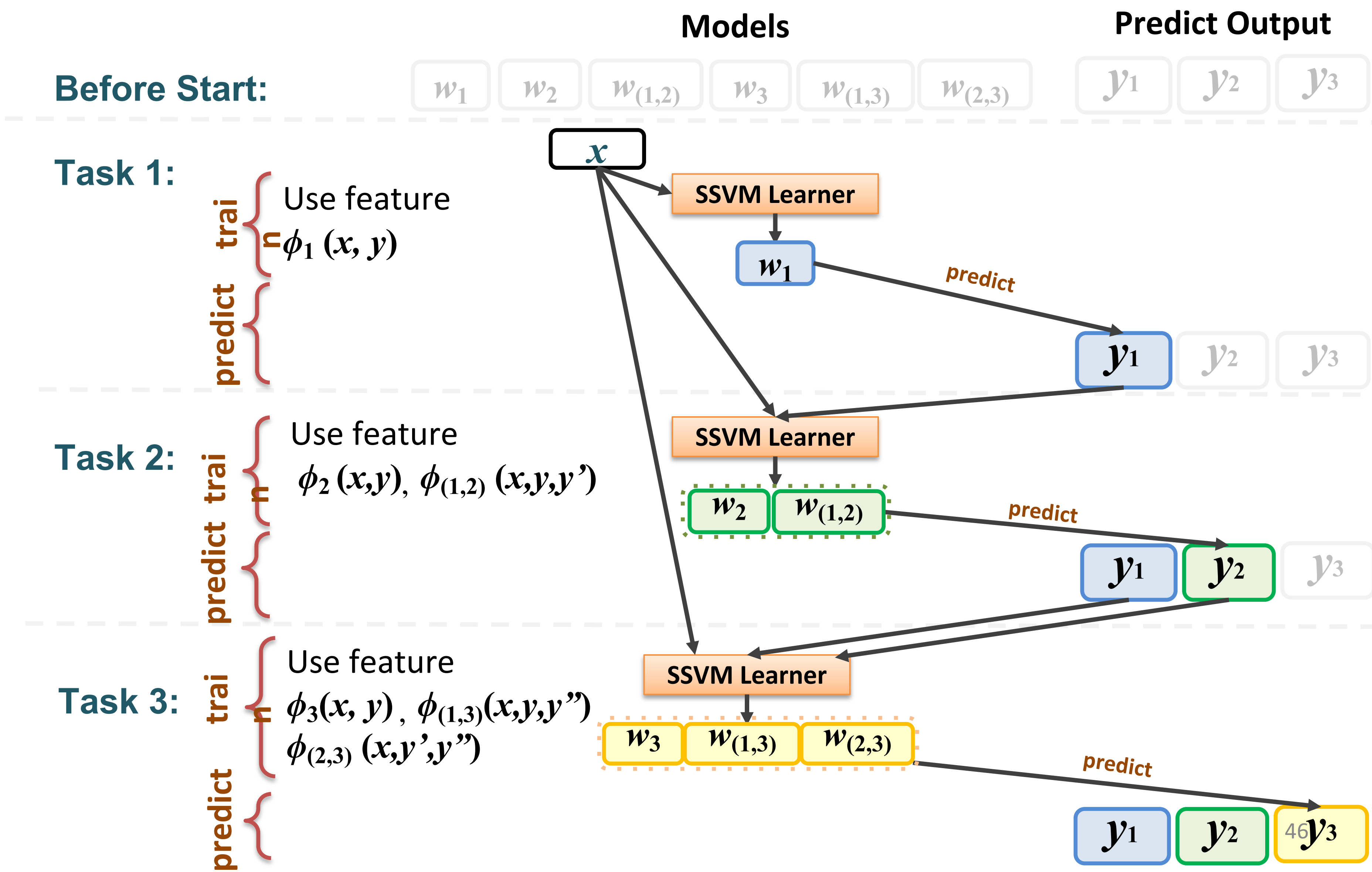
Pipeline Architecture

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



Pipeline Architecture

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



Joint Architecture

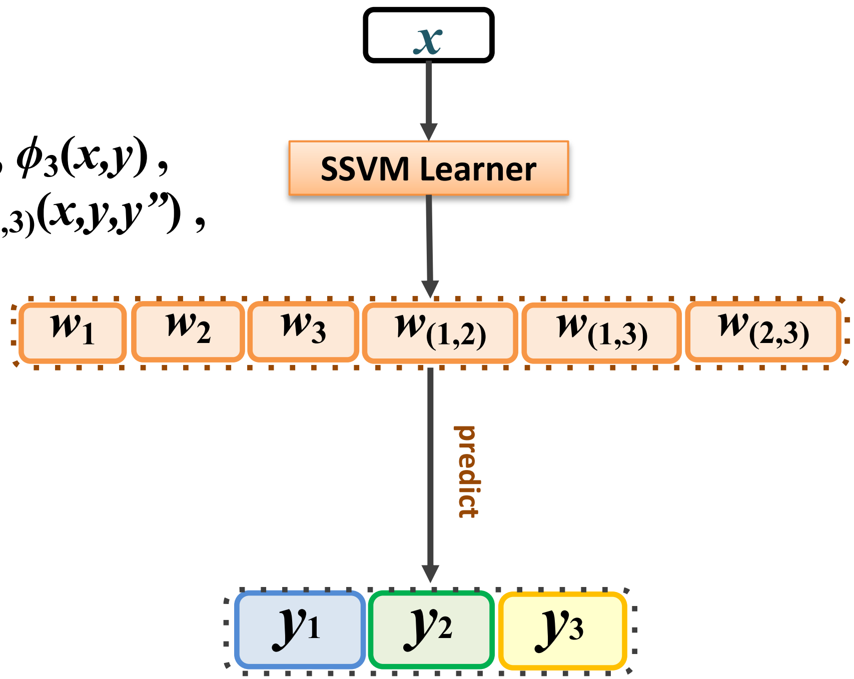
Task 1 & 2 & 3:

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



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

$\circ$ → **Vector concatenation**

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:** y_1 y_2 y_3

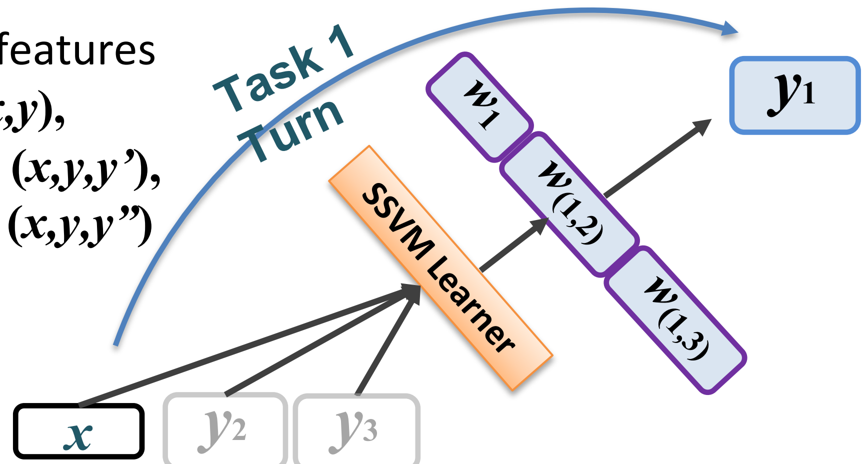
Cyclic Architecture

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

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

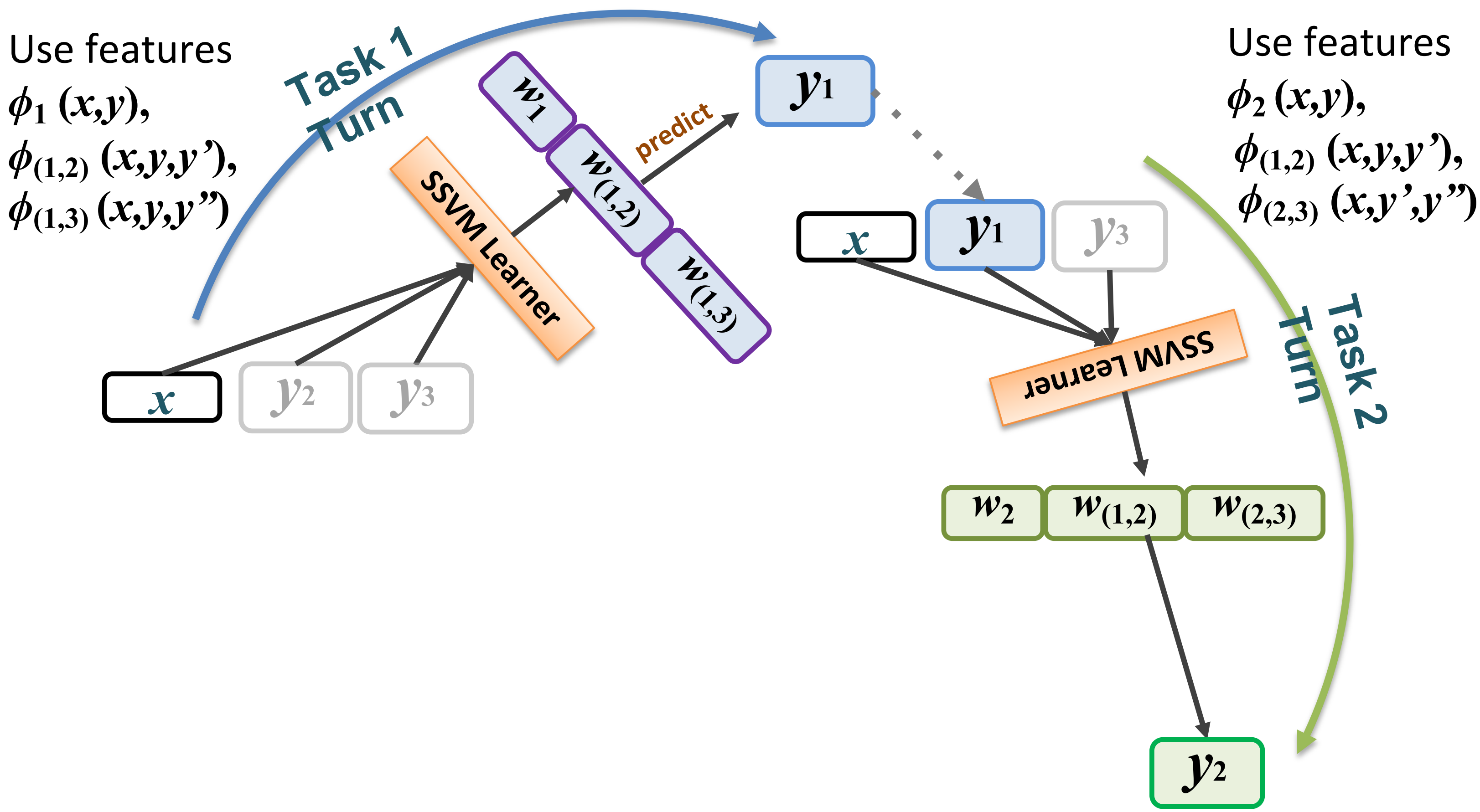


Cyclic Architecture

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

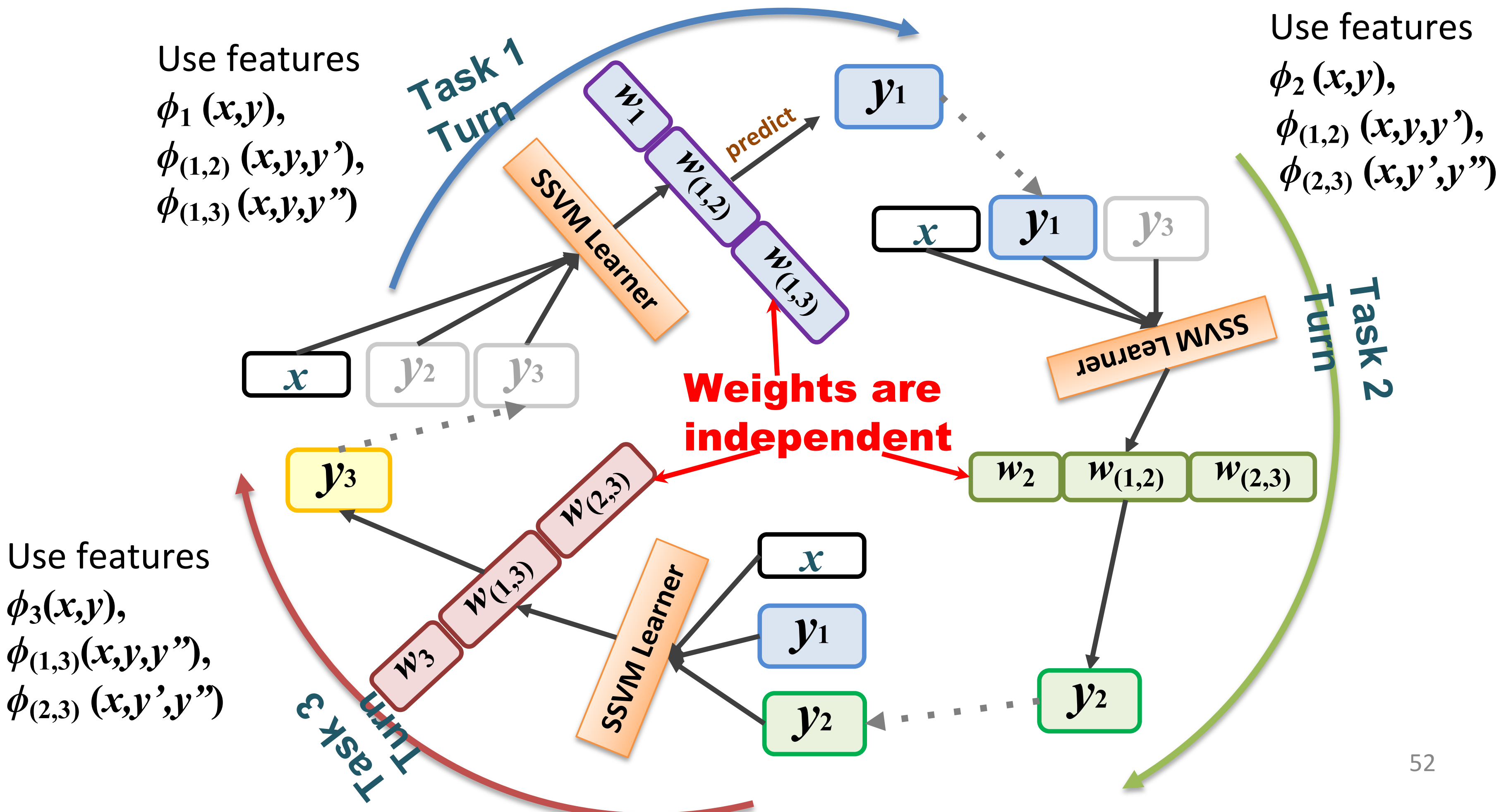


Cyclic Architecture

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



Experimental Setup

Datasets:

	ACE2005	TAC-KBP2015
	ACE-to-Wiki annotation	
Train/Dev/Test	338/144/117	132/36/167

Knowledge Base:

ACE2005	TAC-KBP2015
Wikipedia (2015 dump)	**Freebase** (2014 dump)

Evaluation:

MUC
BCube
CEAF*e*
average → CoNLL

Coref.	*NER*	*Linking*	*Within.Coref*	*Cross.Coref*	*NER & Linking*
CoNLL	Hamming	Hamming	CoNLL	CEAF*m*	NERLC

Combined accuracy of NER and Linking

All metrics are accuracies (larger is better)

Results

Cyclic 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
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							
Unshrd-Wt-Cyclic	81.83	76.05	73.99	77.29	84.18	80.67	11min

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
c. Results of Cyclic Architecture						
Ushrd-Wt-Cyc	89.57	77.68	74.6	82.08	80.5	3m52s

- Competitive accuracy, and much faster training

Summary of MTSP for Entity Linking

1. Formulated the problem of multi-task structured prediction (MTSP) in the context of entity analysis of NLP.

1. Developed a search-based learning framework: structured SVM for training; beam search for inference.

1. Studied three architectures: *pipeline, joint*, and *cyclic* to trade-off between accuracy and speed.

1. Evaluated two pruning approaches for the joint architecture

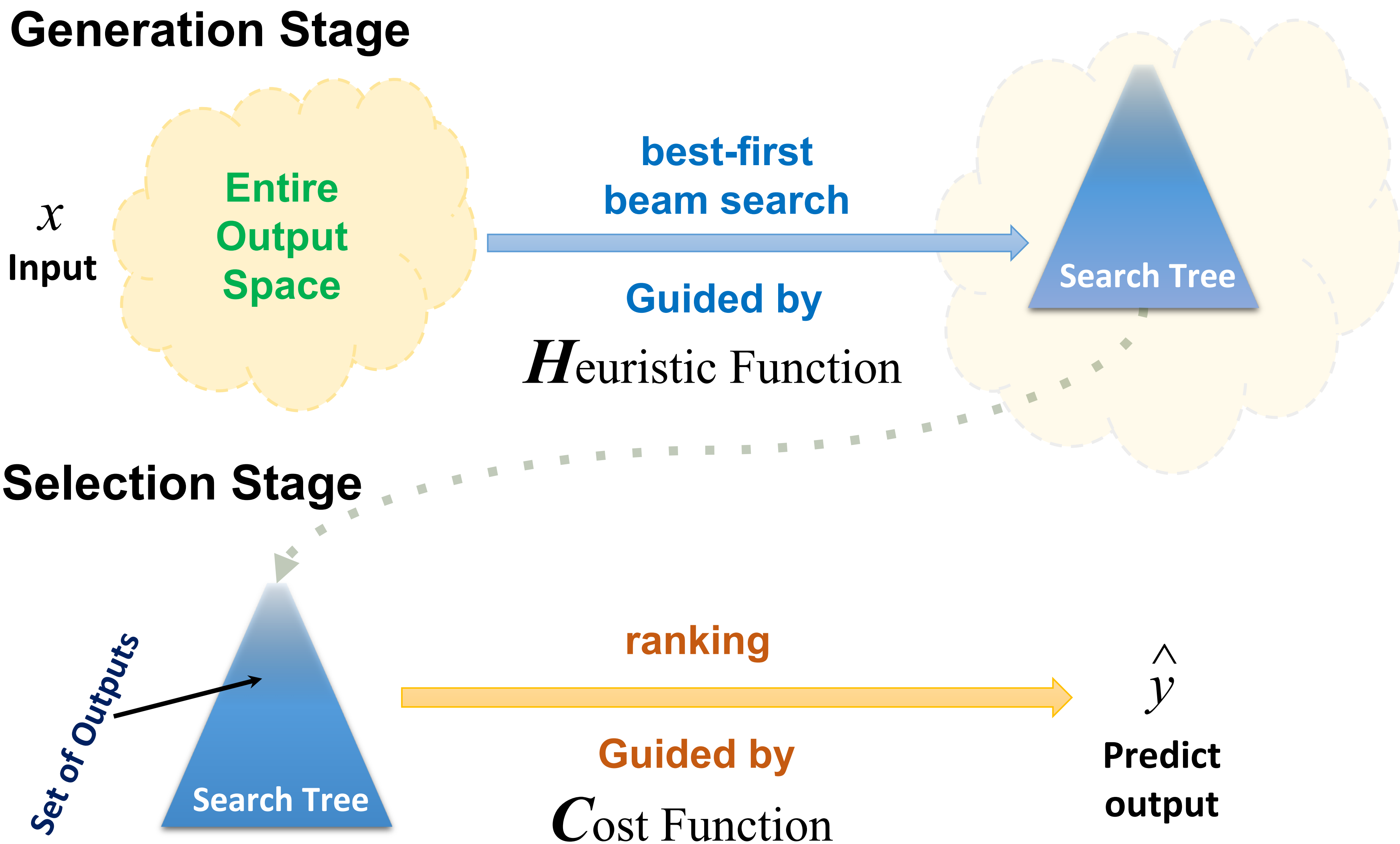
HC-Nets: A Framework for Search-based Deep Structured Prediction

Prior search-based structured prediction approaches use hand-coded features.

We develop a general search-based framework that can perform **neural network function** learning under the **discrete complete output space**.

The decomposition of heuristic and cost functions makes the representation more expressive and learning more modular.

HC-Nets for Structured Prediction



Generation Stage: H-Search

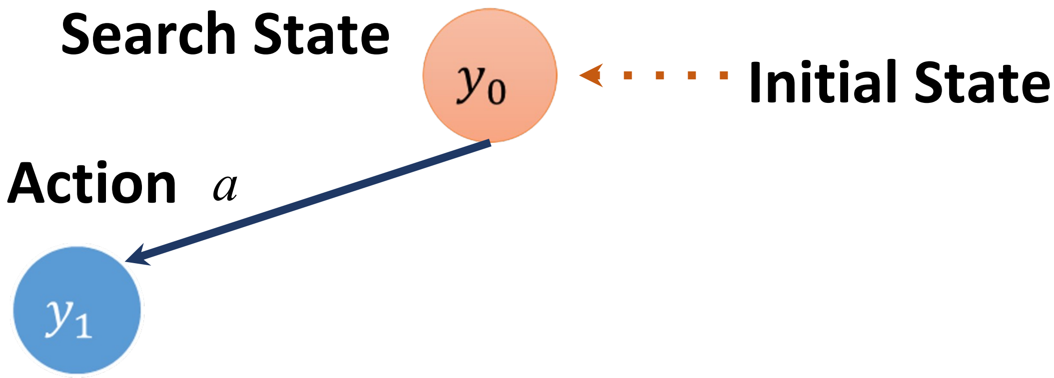
Search State y_0 **Initial State**

Input x

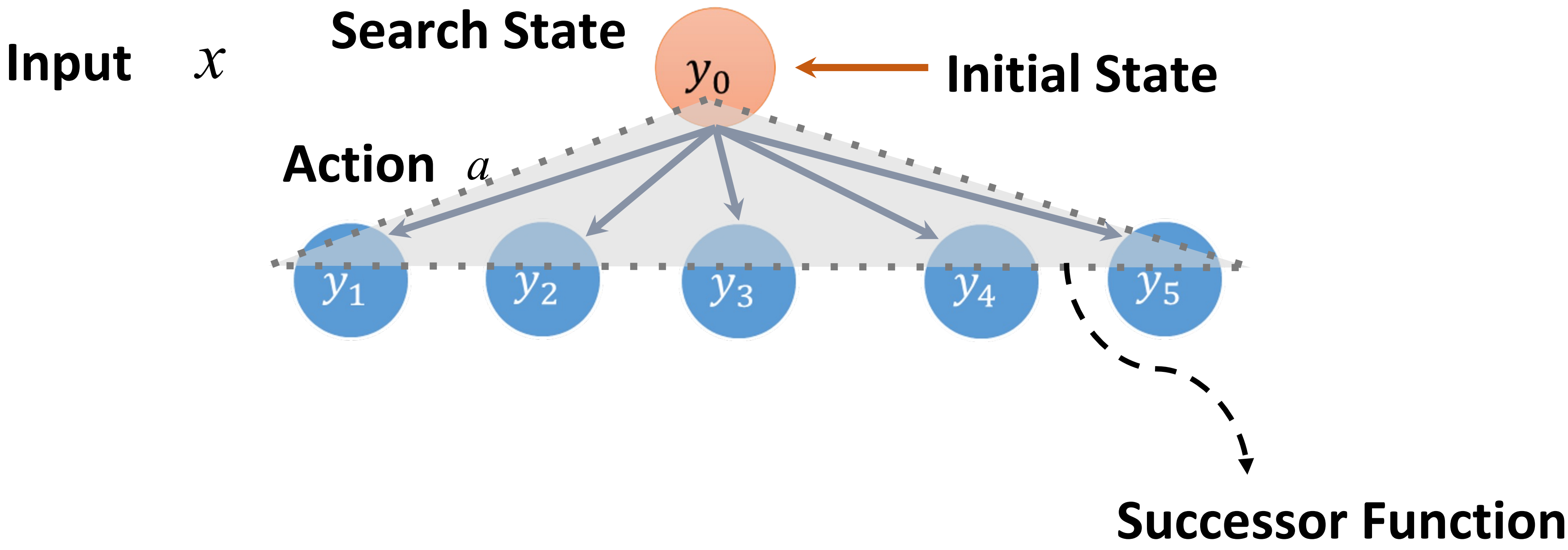
Starting point of search

Generation Stage: H-Search

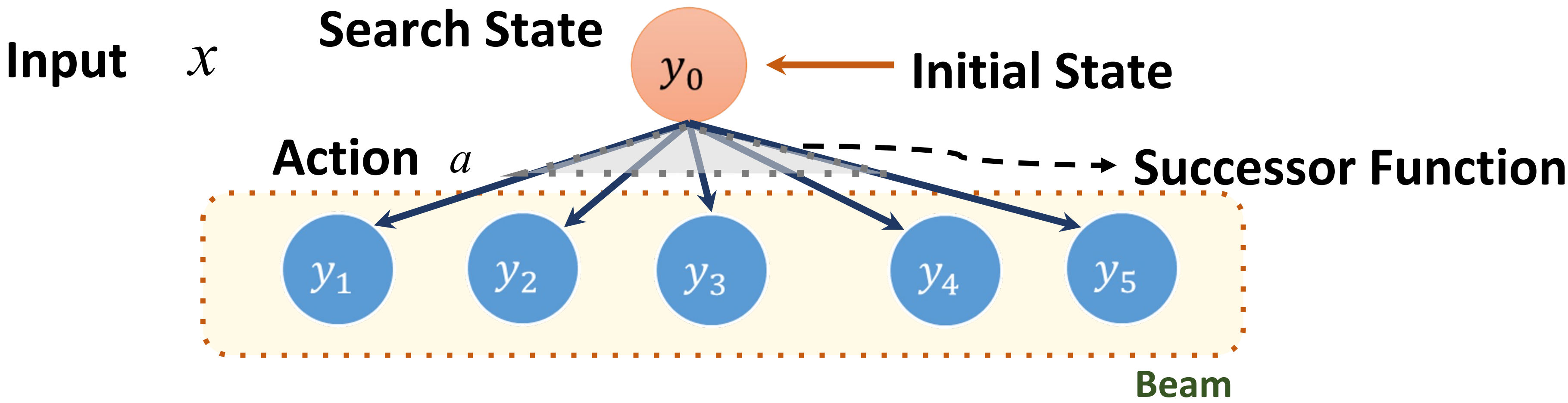
Input x



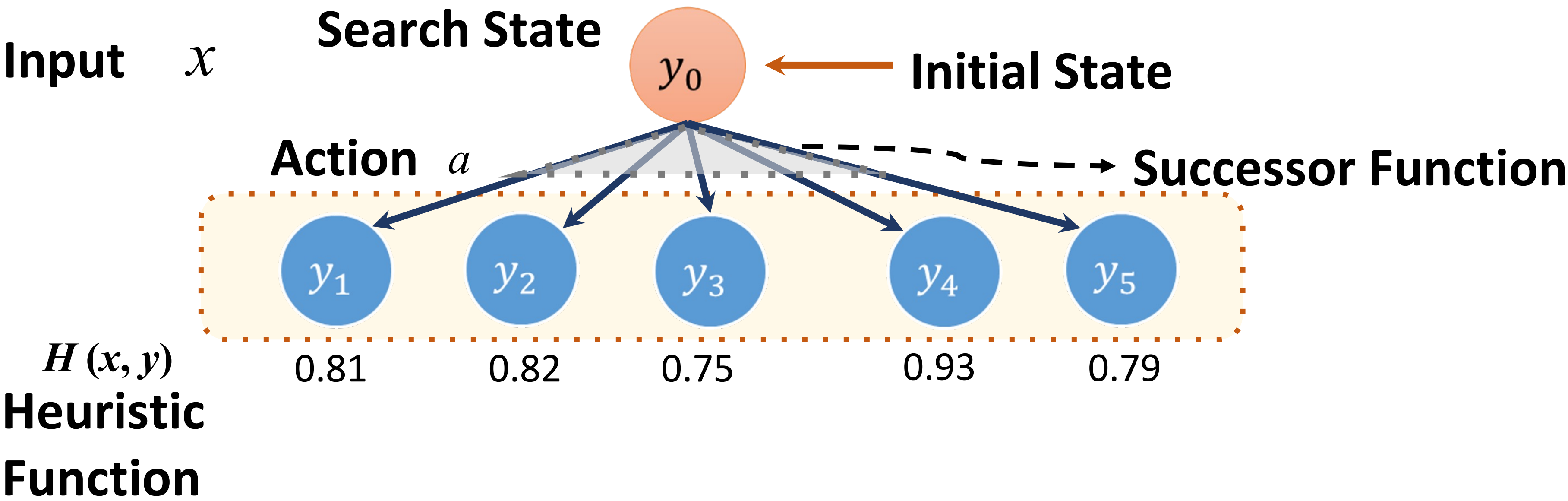
Generation Stage: H-Search



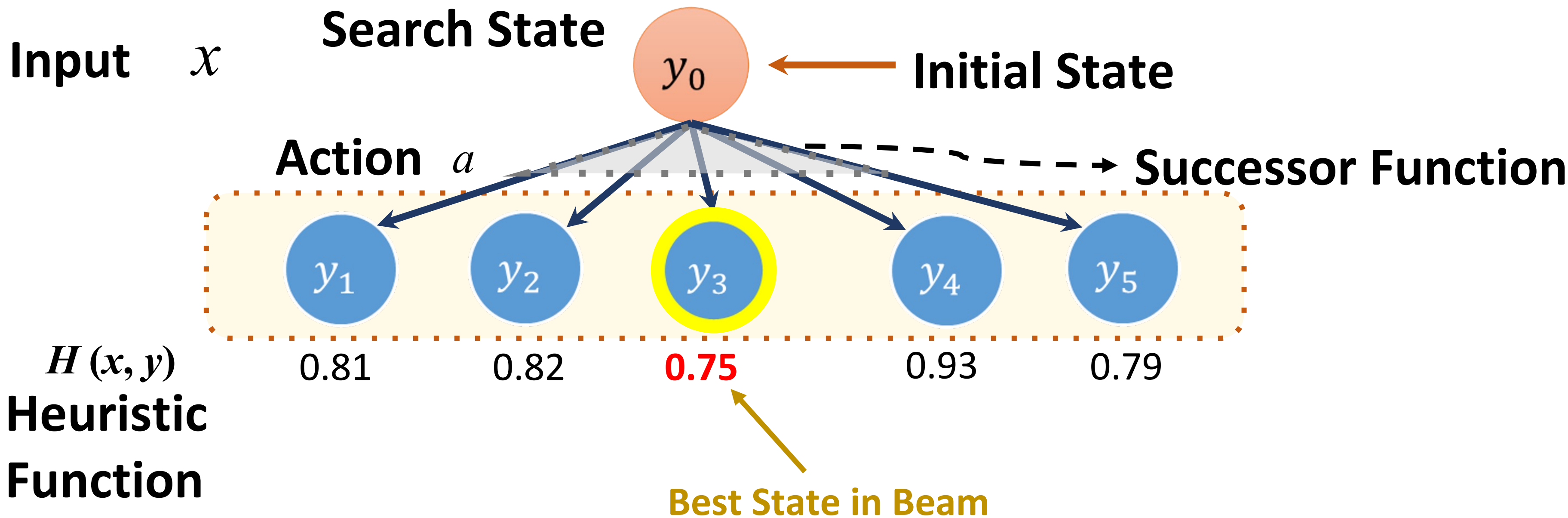
Generation Stage: H-Search



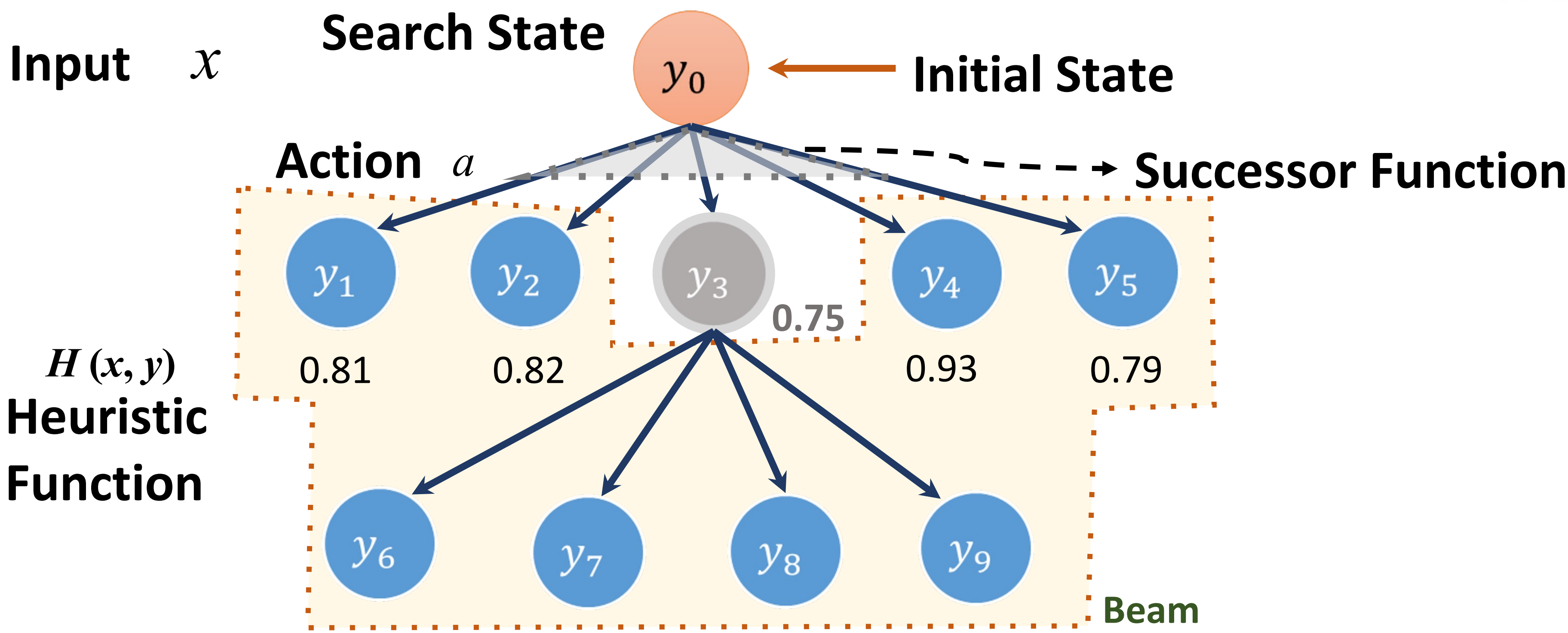
Generation Stage: H-Search



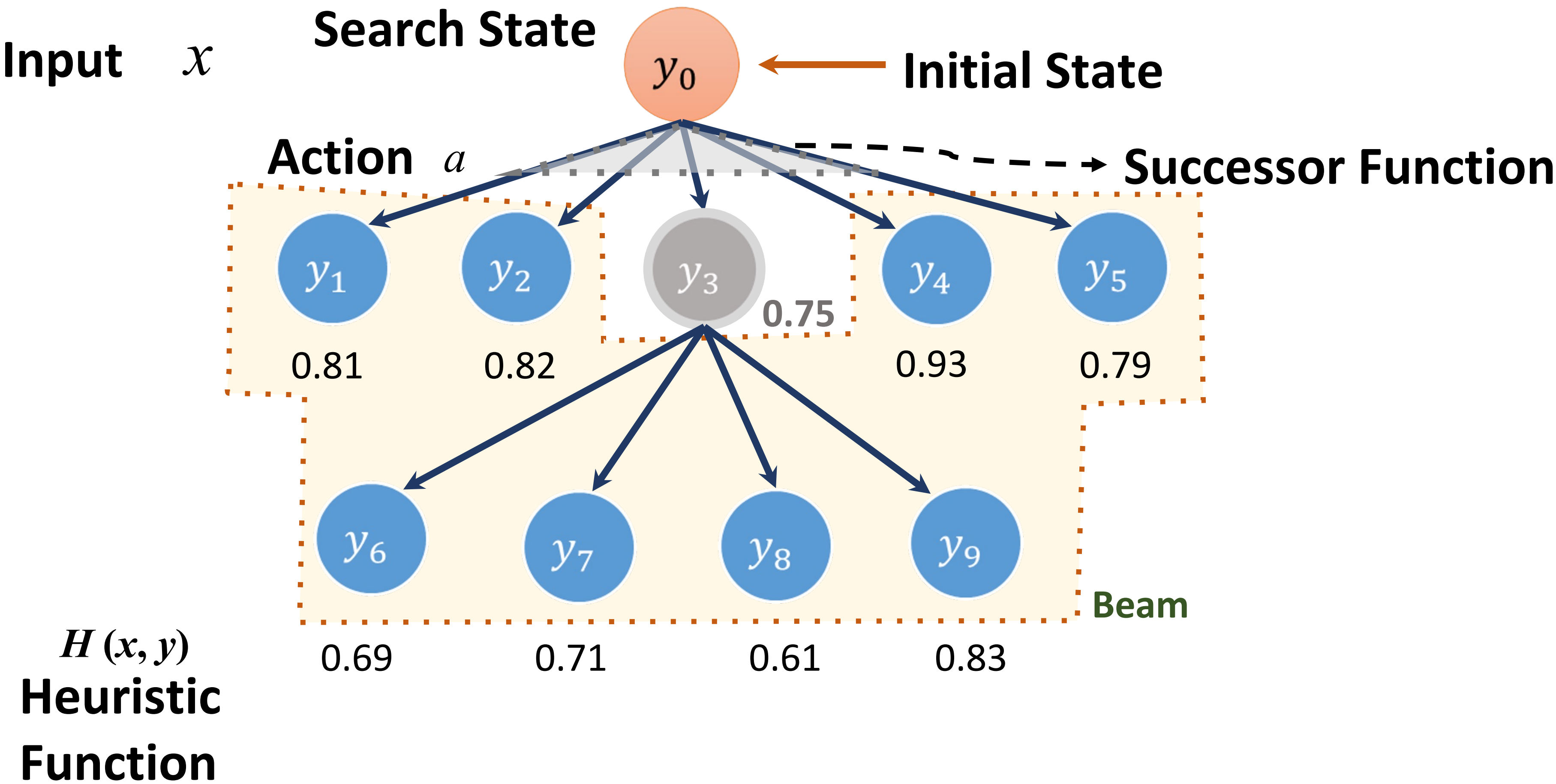
Generation Stage: H-Search



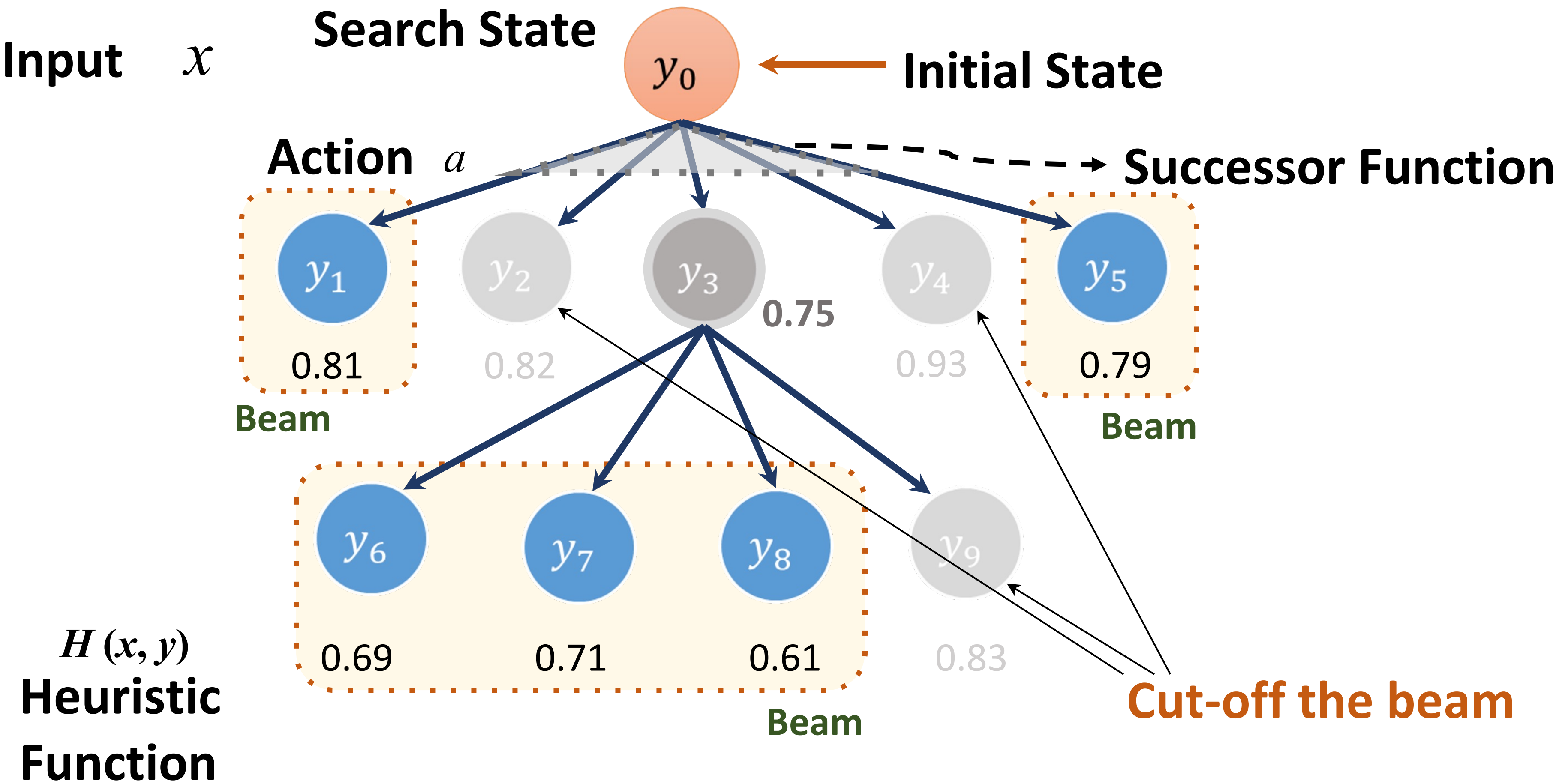
Generation Stage: H-Search



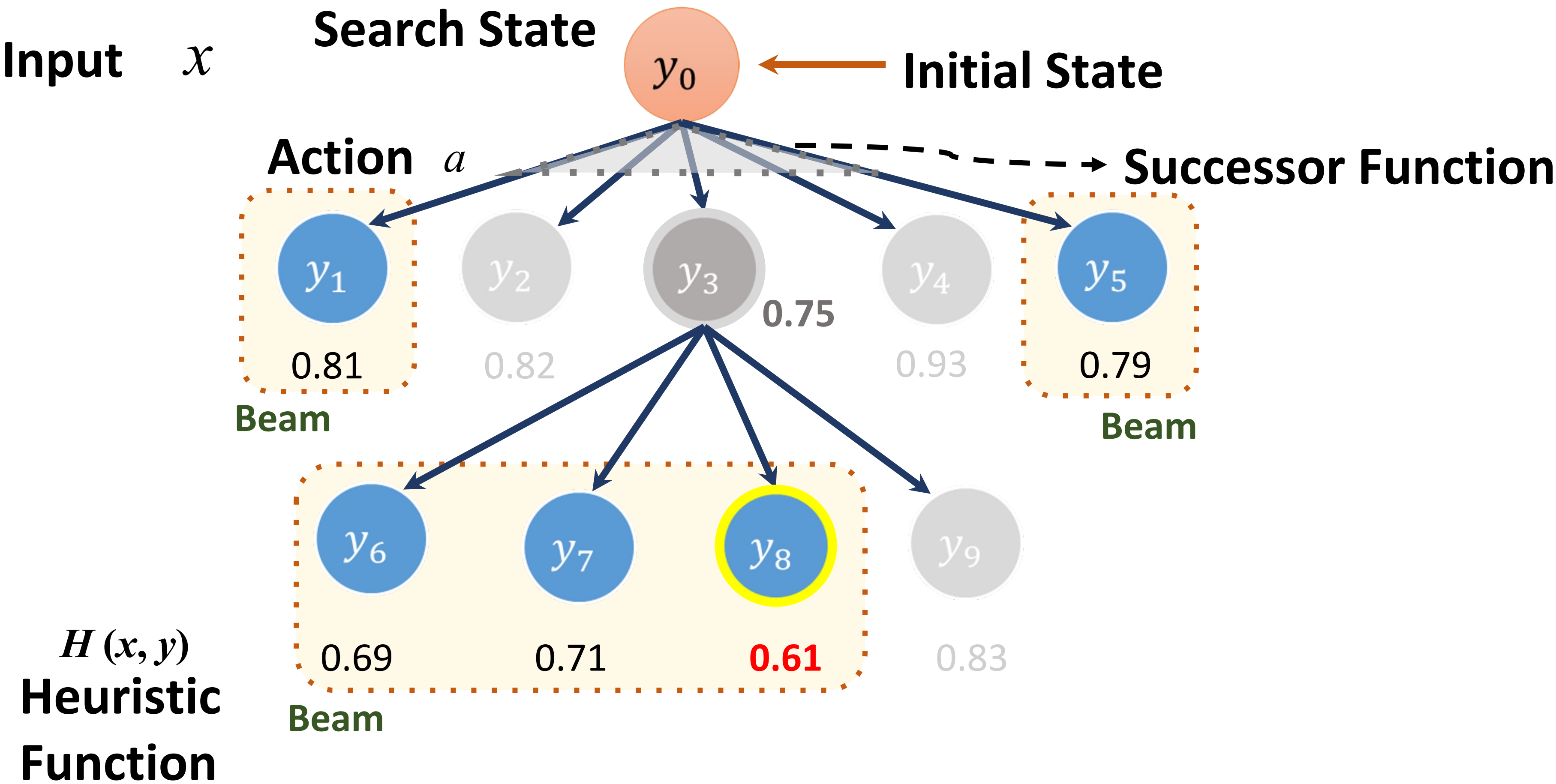
Generation Stage: H-Search



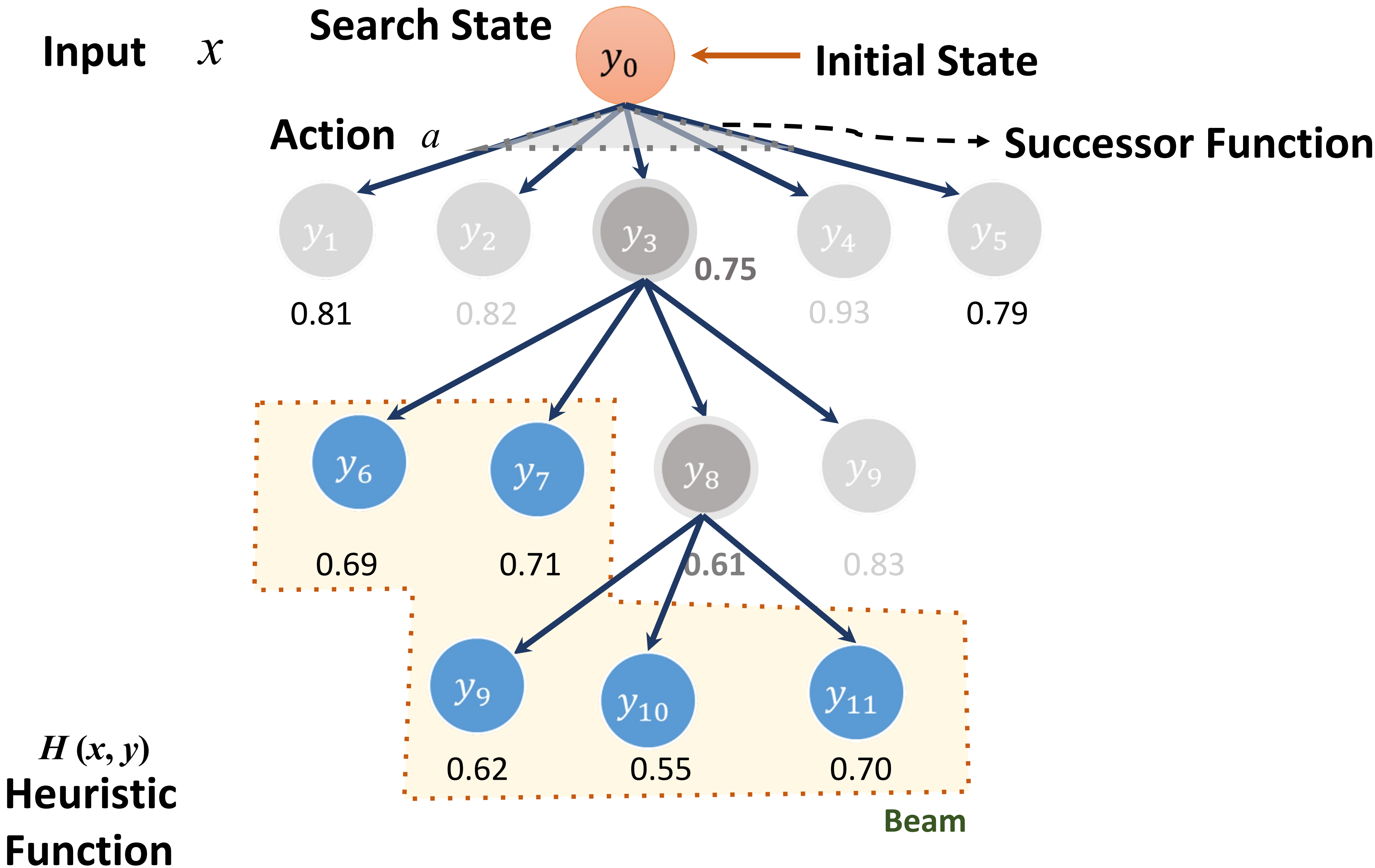
Generation Stage: H-Search



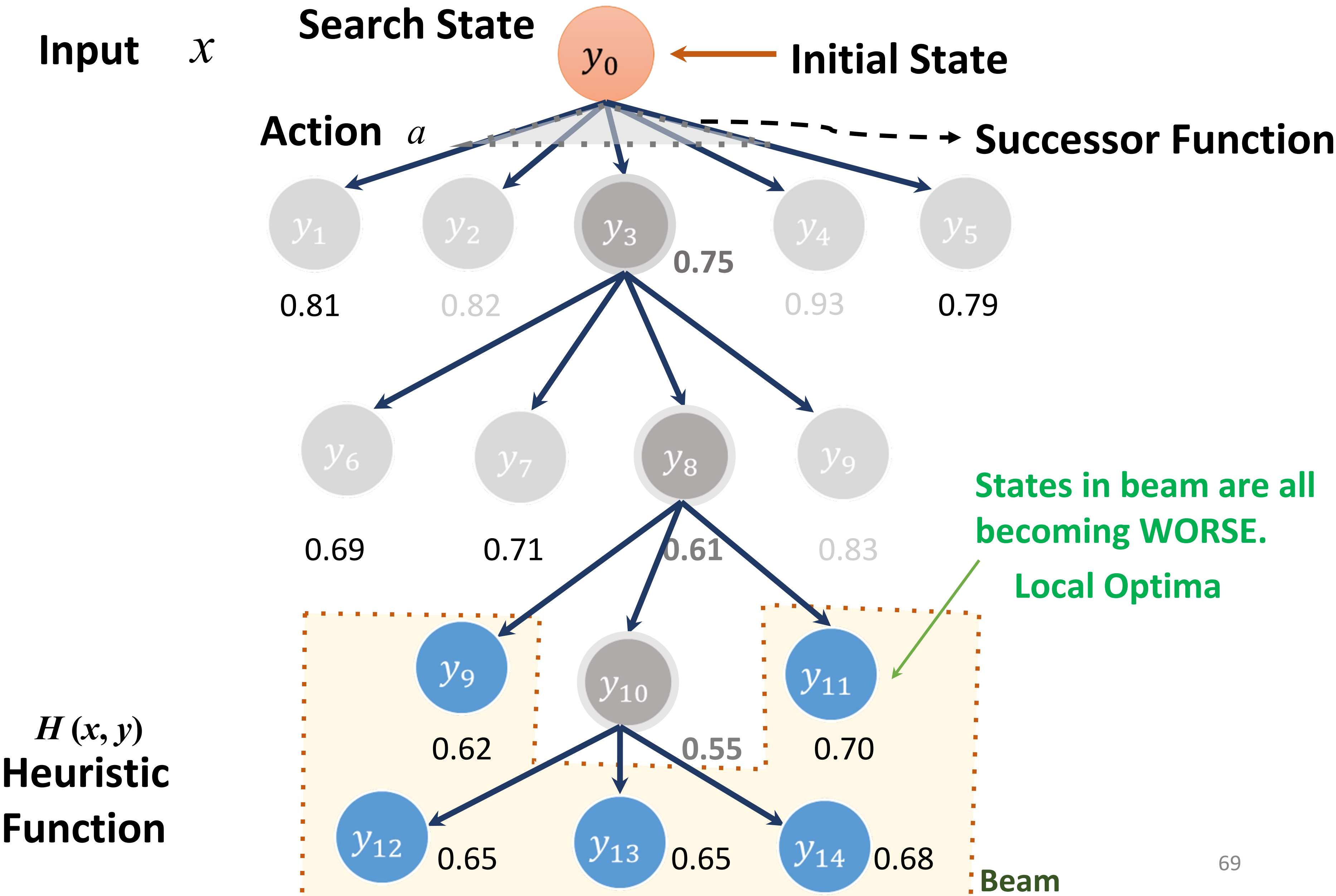
Generation Stage: H-Search



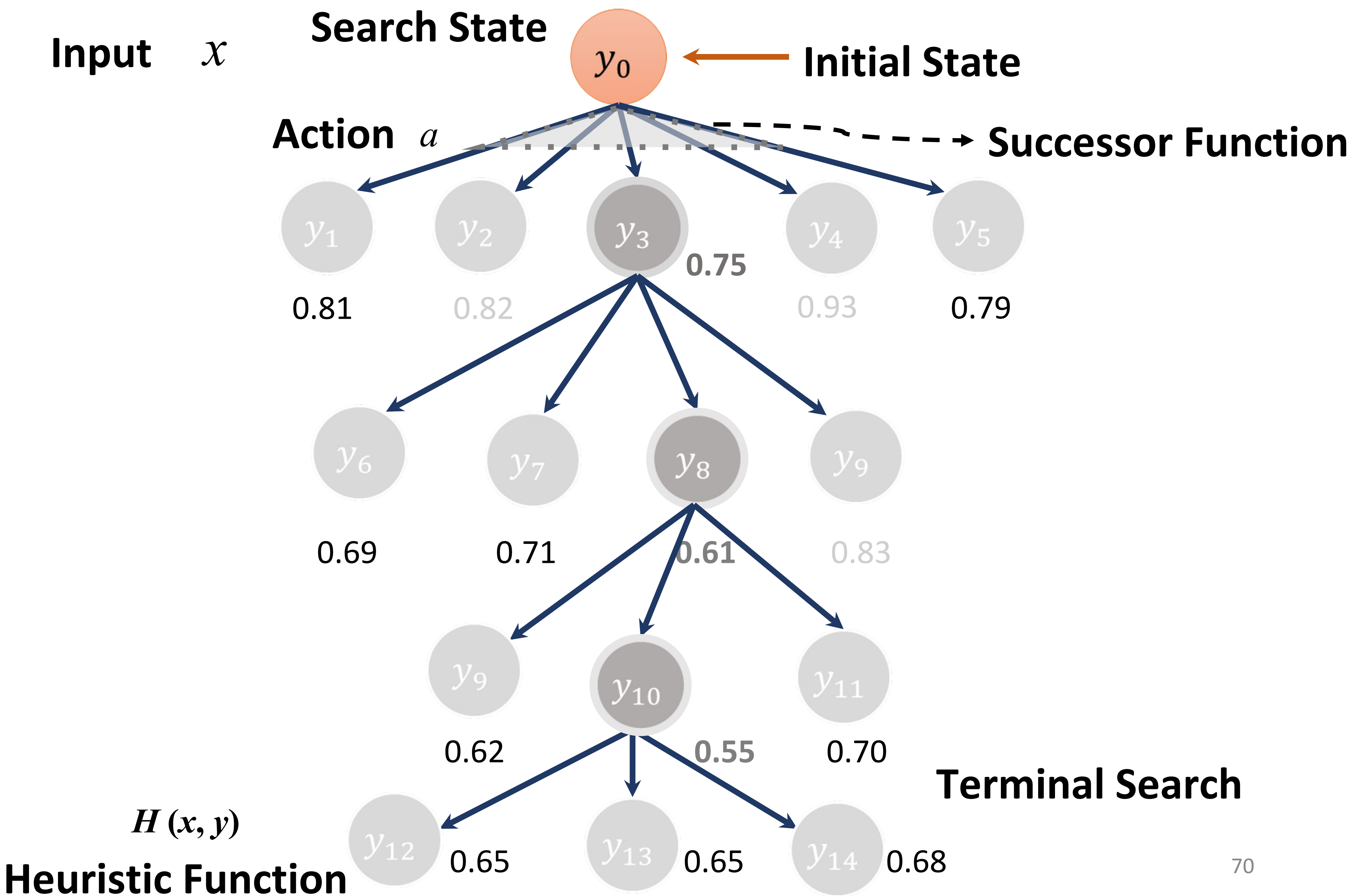
Generation Stage: H-Search



Generation Stage: H-Search

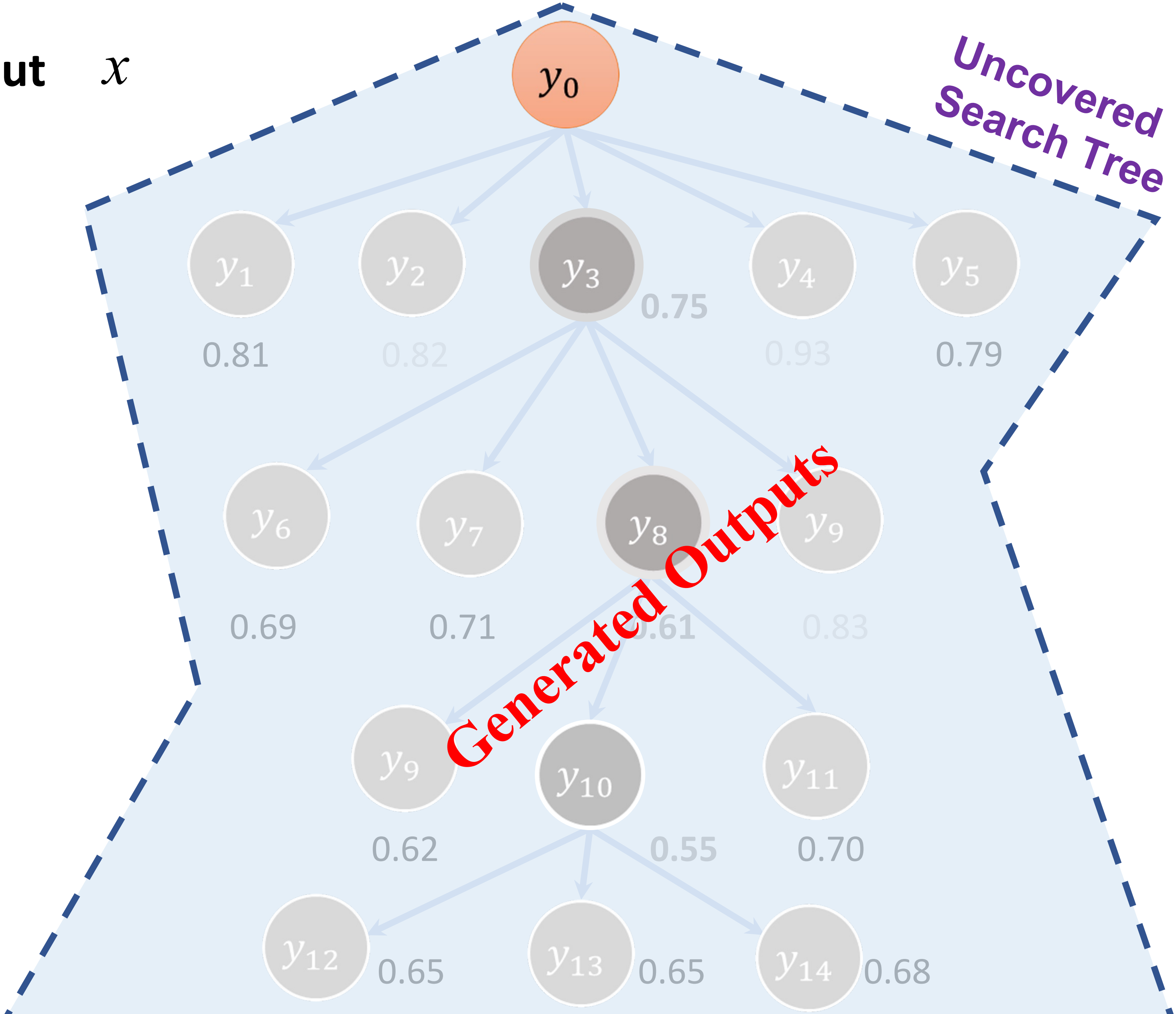


Generation Stage: H-Search

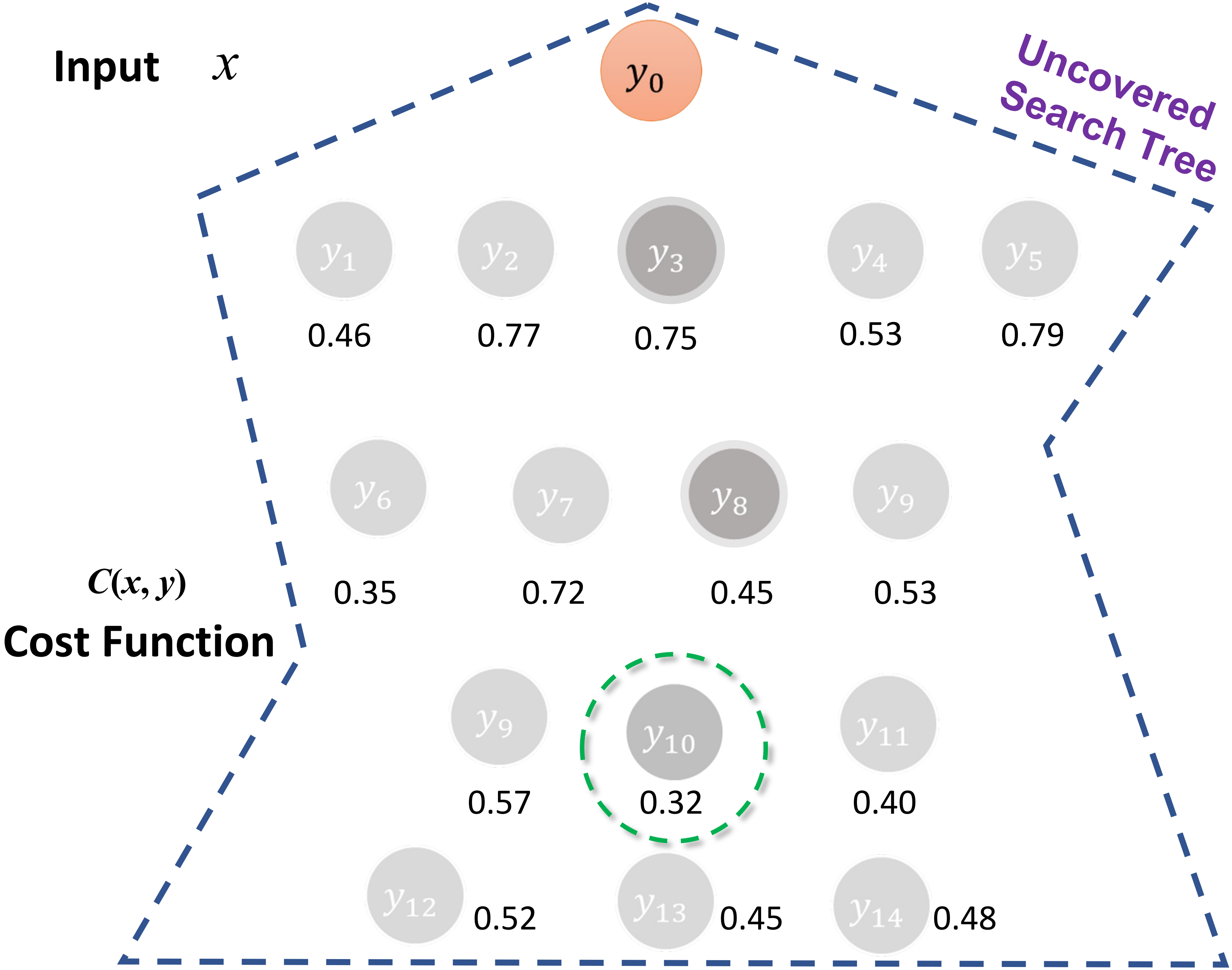


Generation Stage: H-Search

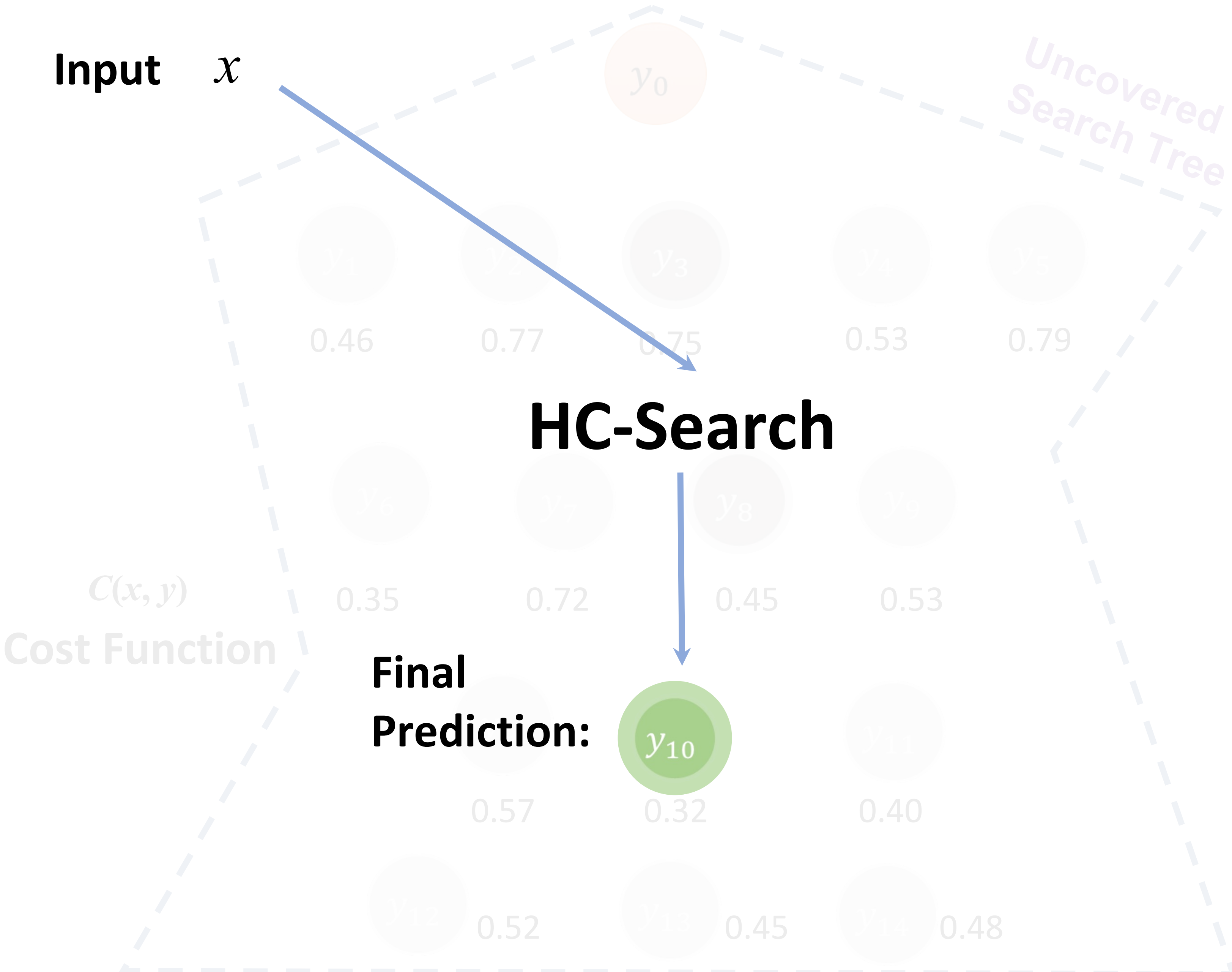
Input x



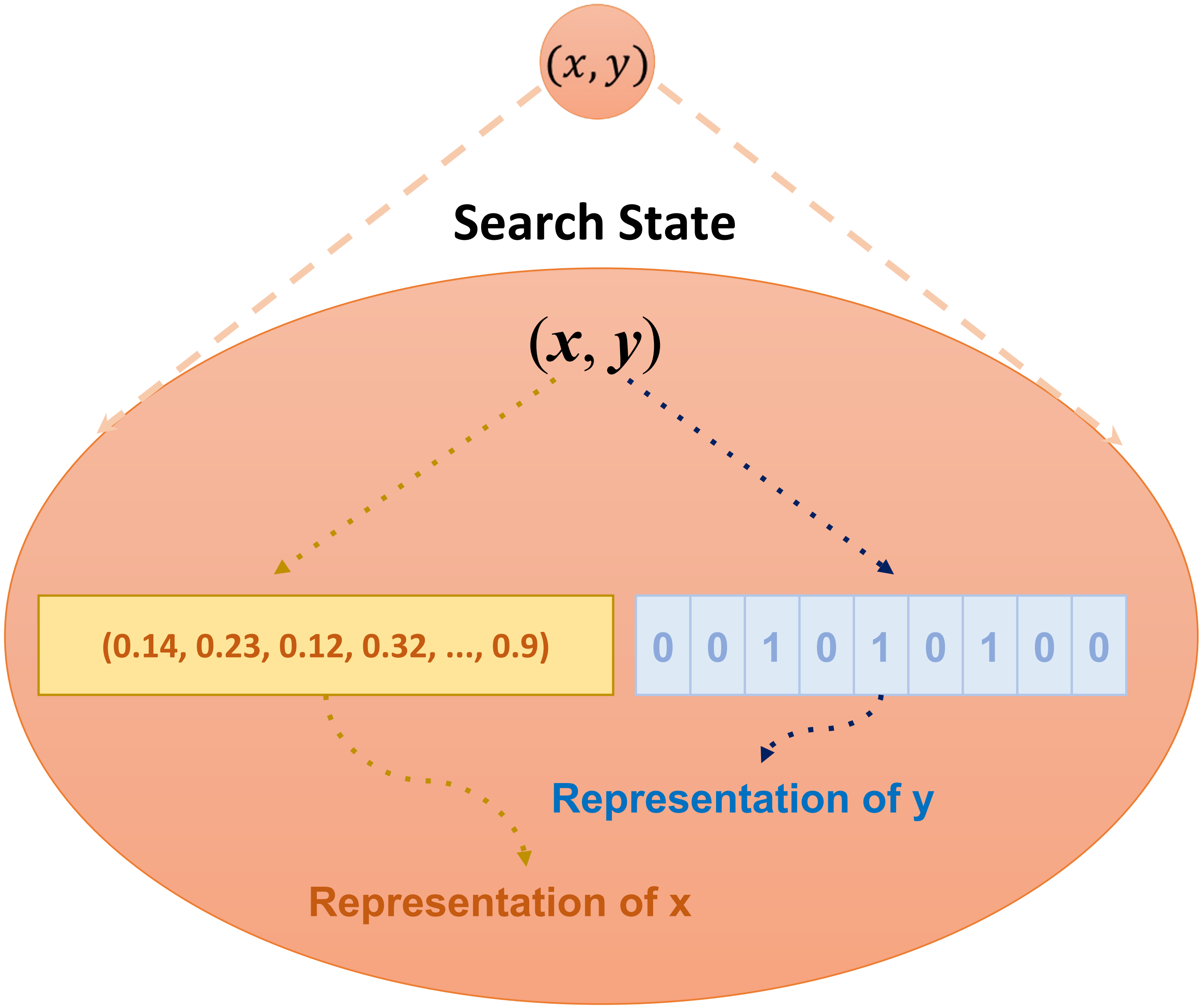
Selection Stage



Selection Stage



Search Space Design



Search Space Design

Search state. Each state contains an input output pair (x, y)

$\boldsymbol{x}$ representation: original features vector

$\boldsymbol{y}$ representation: concatenation of T one-hot vectors

concatenation

State representation

Example:

y = 1 2 1 0

y representation: 0 1 0 0 0 1 0 1 0 1 0 0

Continuous representation vs Discrete representation?

Contininious representation: richer input information

Discrete representation: (a) no need rounding threshold;
(b) easy to define hard constraints.

Search Space Design

*We will apply **k-Flipbit** search space, an extension of standard Flipbit space.*

Actio
n.

$(y_i \rightarrow z)$: change y_i to a new value z

Original Output:

1	2	3	4	5	6	7	8	9	10	11	12
1	0	1	0	1	0	0	1	1	1	0	1

Action: $a = \{\ (3^{\text{rd}}\text{: } \boxed{1} \rightarrow \boxed{0}\ ),\ (7^{\text{th}}\text{: } \boxed{0} \rightarrow \boxed{1}\ ),\ (10^{\text{th}}\text{: } \boxed{1} \rightarrow \boxed{0}\ )\ \}$

New Output:

1	0	0	0	1	0	1	1	1	0	0	1

Successor
Function.

Branching Factor: $$\sum_{k'=1}^{k} \binom{T}{k'} \times (n-1)$$

At most *k* variables

Choose *k'* variables to make change

Alternative values

Search Space Design

Initial State.

- Use **random initial states** to avoid the overfitting for a particular initial state.
- Use **learned initial states**, where we use a learned I.I.D. classifier to predict each output bit independently.

(e.g., Logistic Regressor)

Terminal State.

- Complete output space search have no hard criteria of terminal state.
- A common condition to stop: reaching a **locally optimal state** or reaching **the maximum depth limit**.

H and C networks are usually ***task-specific***.

$$f: X \times Y \rightarrow R^{+}$$

Input: the input-output presentation pair

Output: a real-value score

Stage-wise Learning for H and C

Heuristic Function Training

Goal: **Uncover** a set of candidate "**high quality**" outputs.

$$S \xleftarrow[\text{Search Tree}]{\text{Outputs in}} \textbf{Beam-Search}(H(x,y;\theta_H))$$

Update:

$$\text{new} \quad \theta_H \longleftarrow \max_{\theta_H} \boxed{\min_{y \in S} \sum l(x,y,y^*)}$$

Defined by the best-loss output in generated set

Cost Function Training

Goal: Optimize over the best cost output among candidate set.

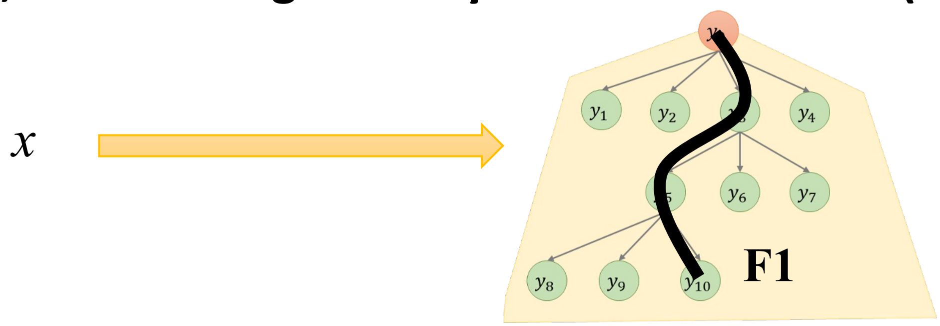
$$\hat{y} \longleftarrow \min_{y \in S} C(x,y)$$

Update: $$\text{new} \quad \theta_C \longleftarrow \min_{\theta_C} \sum Error(x,\hat{y},y^*)$$

Heuristic Function Learning

Key steps:

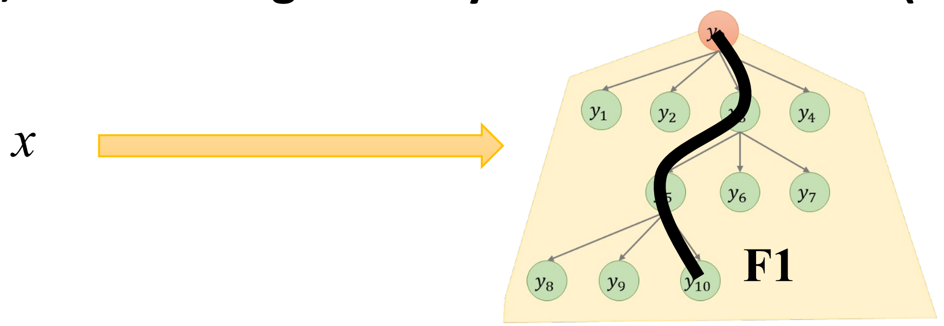
1. **For each input, run search guided by true loss function (e.g., F1)**



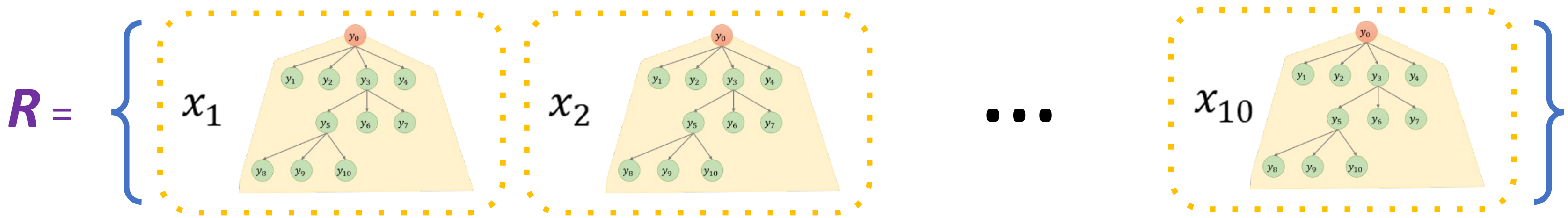
Heuristic Function Learning

Key steps:

1. For each input, run search guided by true loss function (e.g., F1)



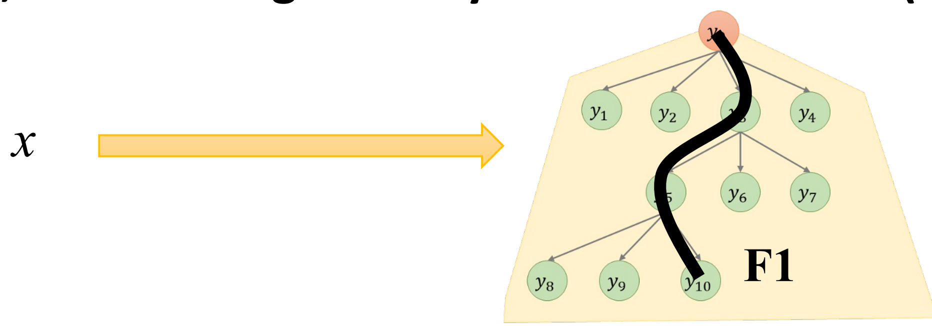
2. Aggregate the uncovered states during search into a set R



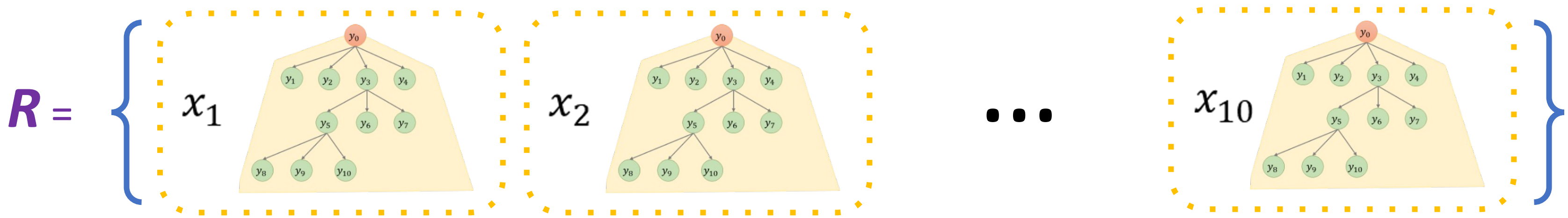
Heuristic Function Learning

Key steps:

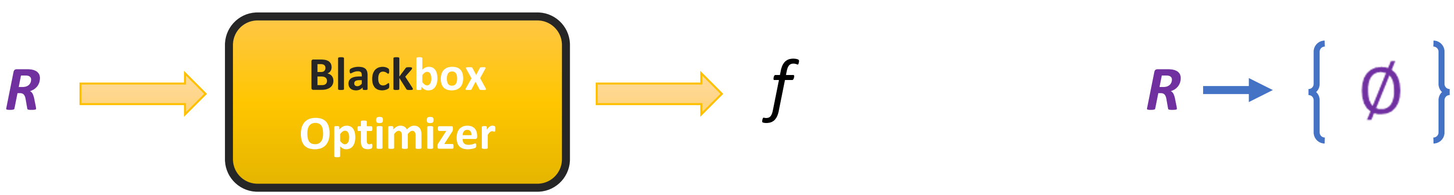
1. For each input, run search guided by true loss function (e.g., F1)



2. Aggregate the uncovered states during search into a set R

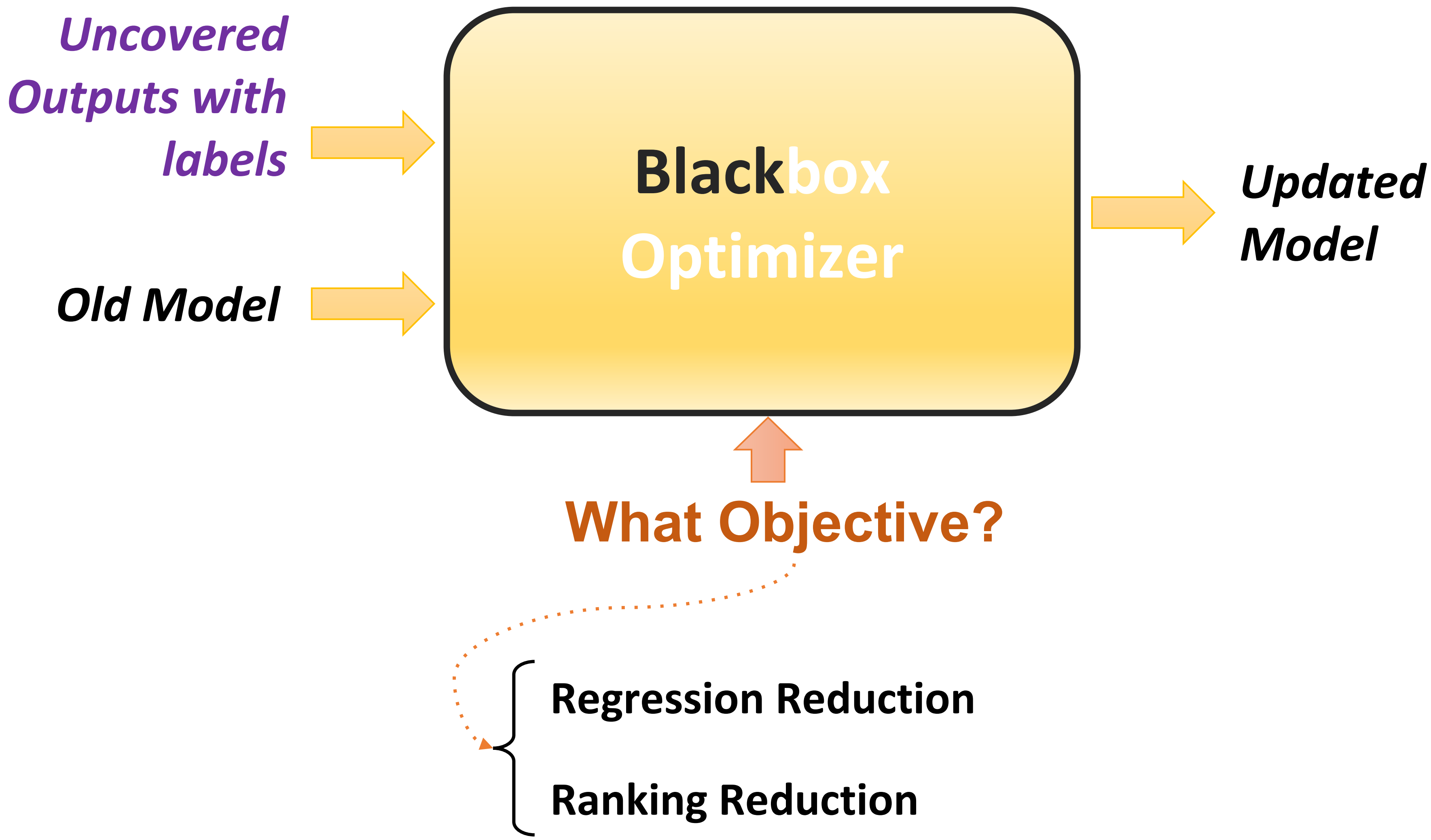


3. After reaching mini-batch examples, sent R into optimizer to do weight update, then clear R



Then Next Iteration…

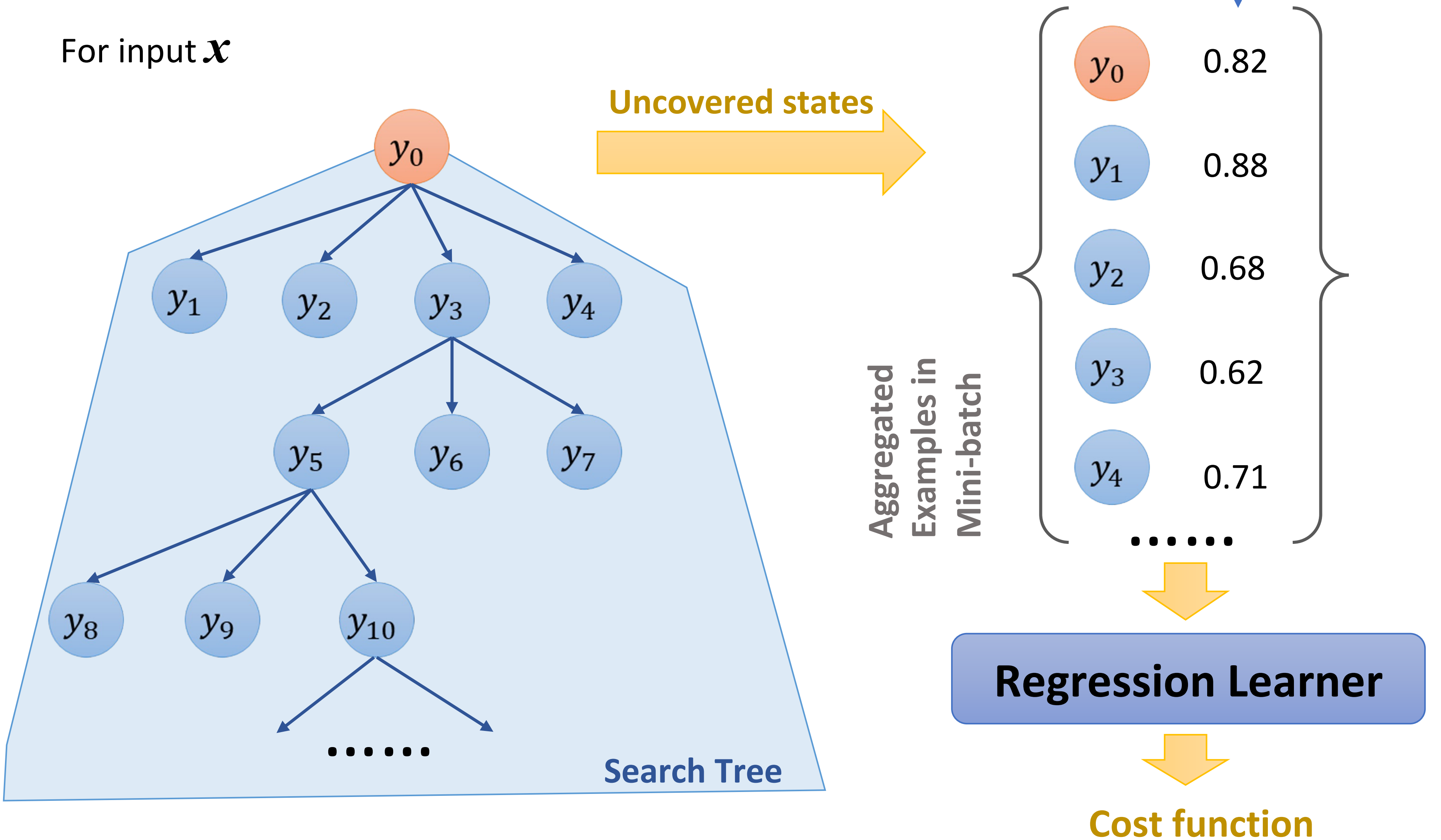
Blackbox Optimizer Reductions



Regression-Based Reduction

Directly estimate the output loss.

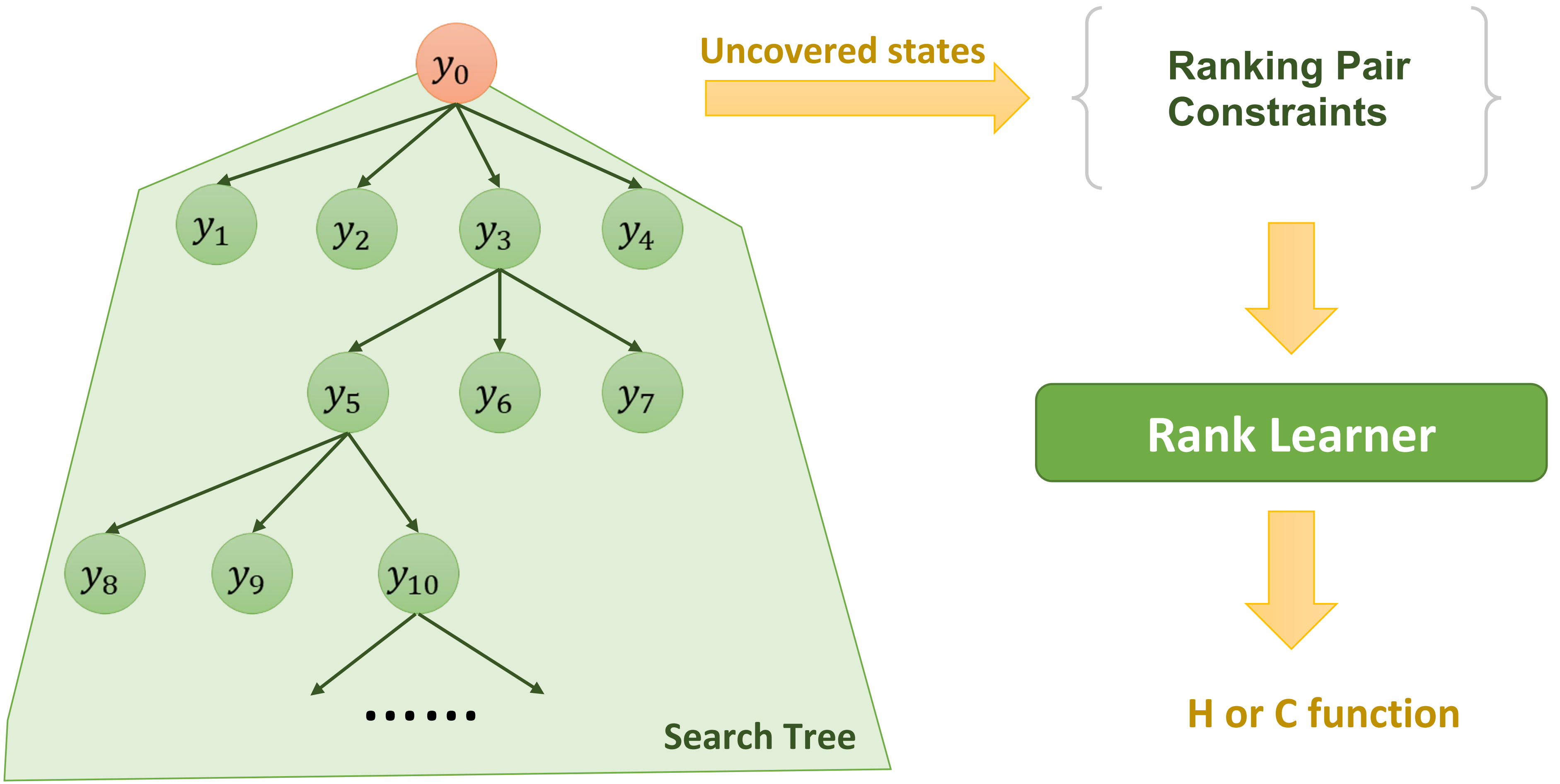
For input $\boldsymbol{x}$



Ranking Based Learner

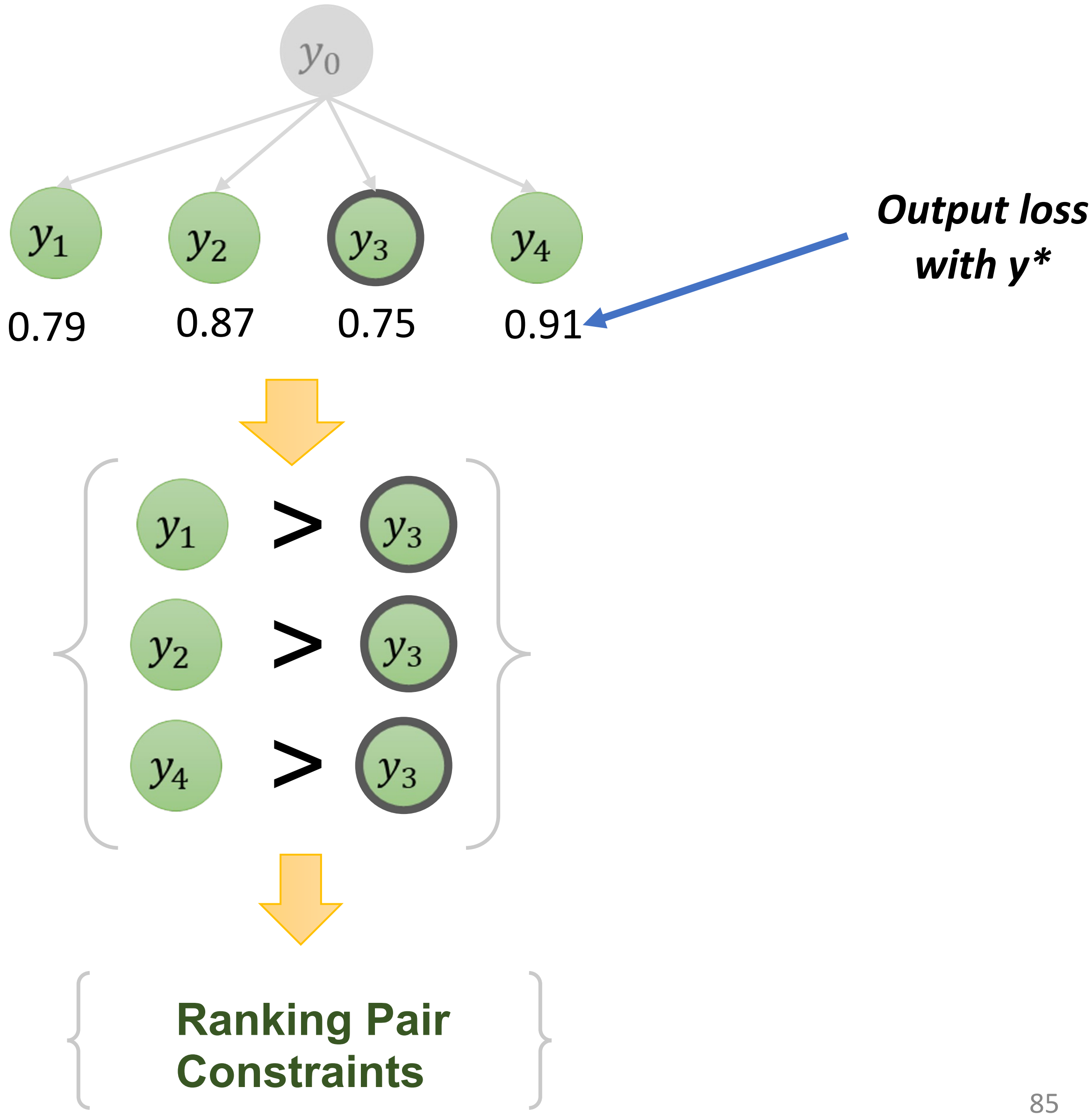
Learning a ranker given ranking constraints.

For input $\boldsymbol{x}$



Ranking Based Learner

Sibling Constraints:



Reduction Summary

Drawbacks of regression-based learning

- First, only the **relative relations** are needed.
- Second, the internal relations of sibling states were not exploited.

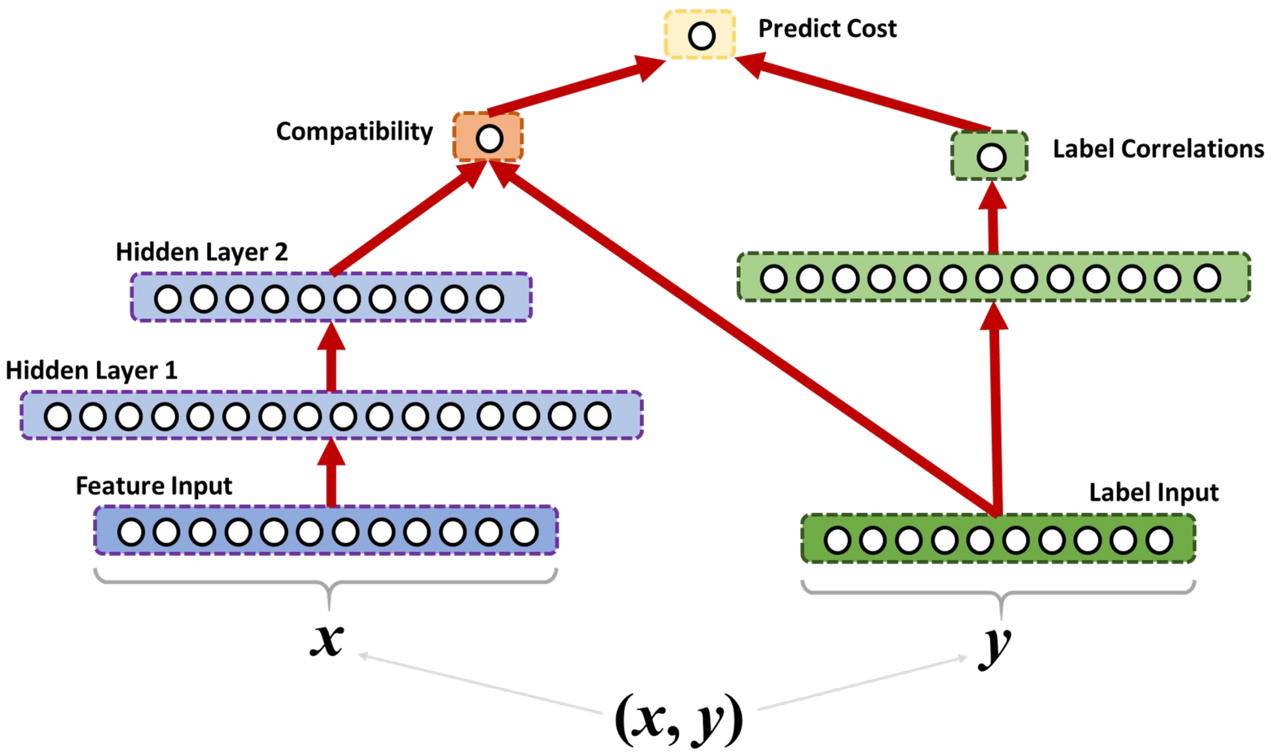
Drawbacks of ranking-based learning

- Training is very computational expensive.

Experimental Setup

Multi-label Classification

- We would evaluate on three datasets: ***Bibtex***, ***Bookmarks*** and ***Yeast***.
- We will report **F1** accuracy.



Our H or C network is derived from **SPEN** and **DVN**

Experimental Setup

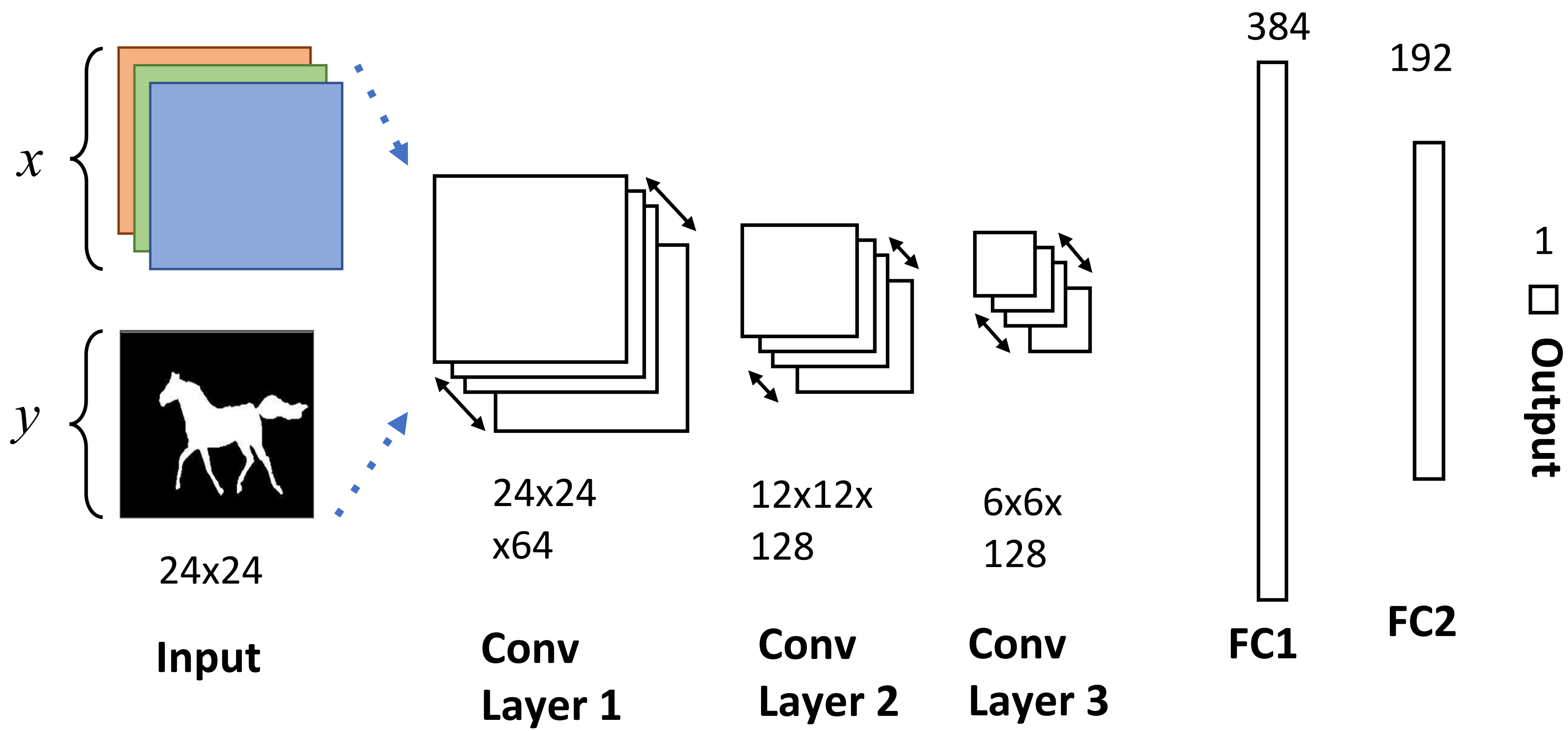
Multi-label Classification

- Maximum number of epoch = 300.
- Randomly split out 5% of the examples as validation set.
- Learning rate, 0.005 for Yeast, and 0.1 for the other three.
- We use the gradient descent optimizer to perform the weight update.

Experimental Setup

Image Segmentation

Evaluate on ***Weizmann Horse*** Dataset. Label Set: {Background = 0, Horse = 1}
Use *Intersection-of-Union (IoU)* and *Pixelwise-Accuracy* to evaluate the result.

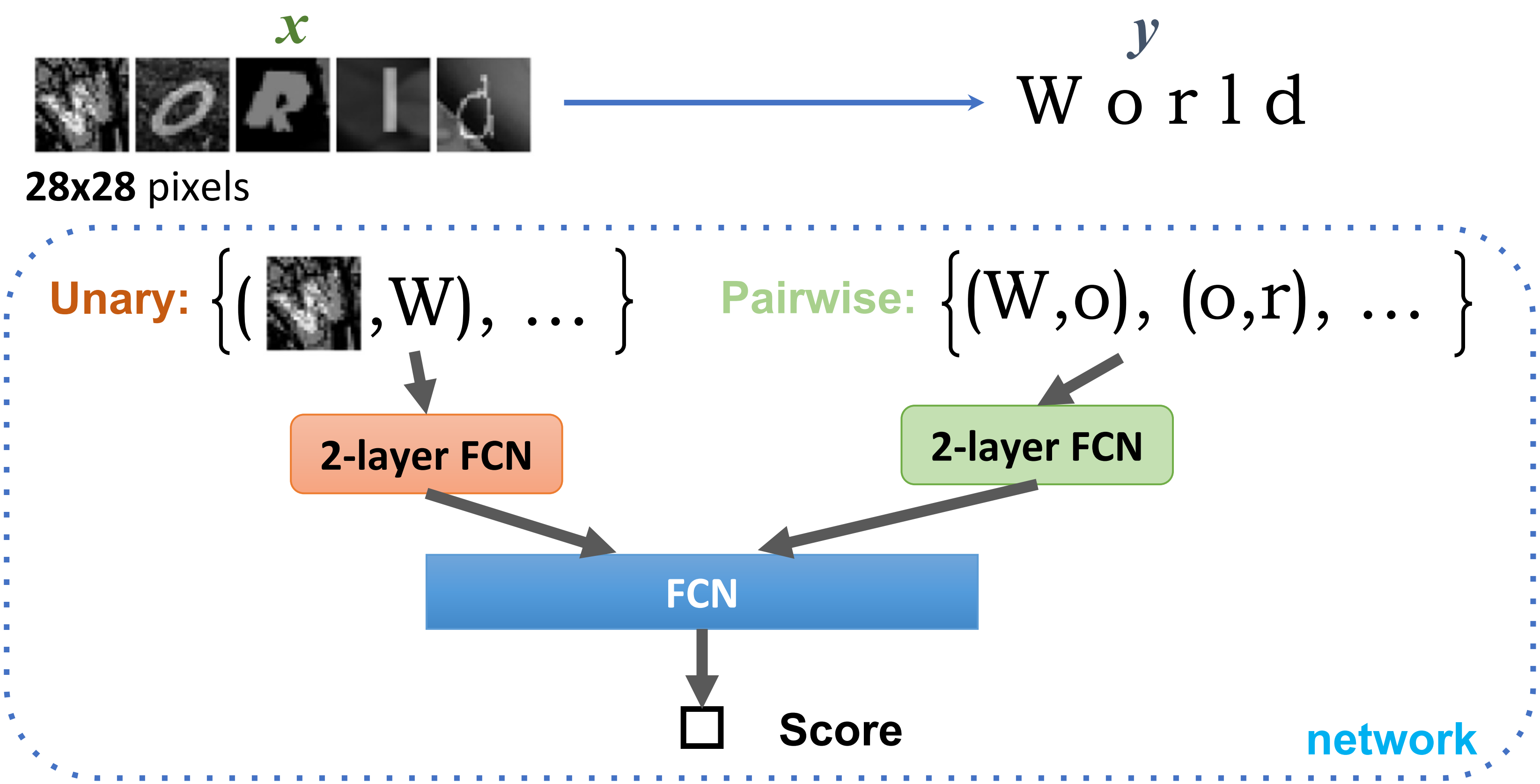


Our cost function network is derived from **DVN**

Experimental Setup

Word Recognition

We use a synthetic word recognition dataset, constructed from ***Char74k*** by taking a list of 50 common five-letter English words.



Evaluate with **character-wise Hamming accuracy**.

Results

Accuracy comparison with HC-Nets and the other SOTA approaches.

	Multi-Label			Word Recog.	Image Segm.
Algorithms	***Yeast***	***Bibtex***	***Bookmarks***	***HW-Words***	***Horse32x32***
	F-1			Char Acc.	IoU
SPEN(E2E)	63.8	38.1	33.9	42.26	75.45
DVN	63.8	**44.7**	37.1	-	**84**
InfNet	-	42.2	37.6	37.95	69.31
NLStruct	-	-	-	44.37	81.86
Rand-Init	62.5	43.2	34.9	39.02	74.62
Learned-Init	**62.6**	**44.7**	**37.8**	**45.14**	82.55

Results

HL-Search

- Analysis of the varying the maximum search depth of generation stage.
- We present the generation and selection accuracy with 2 initialization methods.

Depth	Gen.Acc.		RealAcc.	
	Rand-Init	Learned-Init	Rand-Init	Learned-Init
Yeast				
2	0.531	0.674	0.462	0.579
5	0.755	0.816	0.553	0.623
10	0.816	0.831	0.564	0.627
14	0.854	0.841	0.598	0.629
Bibtex				
2	0.698	0.722	0.312	0.375
3	0.722	0.748	0.384	0.421
4	0.759	0.801	0.385	0.425
5	0.762	0.811	0.385	0.426
Bookmarks				
2	0.791	0.84	0.285	0.344
3	0.791	0.856	0.279	0.357
4	0.792	0.882	0.292	0.358

Depth	Gen.Acc.		RealAcc.	
	Rand-Init	Learned-Init	Rand-Init	Learned-Init
Words Recognition				
1	0.22	0.41	0.15	0.28
5	0.56	0.67	0.32	0.35
10	0.74	0.88	0.38	0.41
15	0.83	0.91	0.37	0.404
Horse32x32				
10	0.24	0.68	0.164	0.531
20	0.57	0.73	0.415	0.628
50	0.79	0.86	0.614	0.719
65	0.88	0.93	0.628	0.698

Future Work

1. **End-to-End MTSP with Deep Neural Networks.**

   How to **formulate** the problem? **What tasks** can be solved jointly? How to **trade off** the efficiency and model complexity?

2. ***HC*-Nets Learning for Variable Length Sequences.**

   How to form a fixed length representation for the non-fixed length input-output pair?

   Variable Length Input (e.g., sentence) $\xrightarrow{?}$ (Fixed Length Representation)

3. **Joint Learning of *H* and *C* networks in *HC*-Nets.**

   Stage-wise training does not follow the end-to-end learning principle.

   $Error_H$ $Error_C(H)$ $\xrightarrow{?}$ $Error_H(C)$ $Error_C(H)$

   Stage-wise *Joint*

Publications

- ***Prune-and-Score: Learning for Greedy Coreference Resolution.***

  Chao Ma, Janardhan Rao Doppa, Xiaoli Fern, Tom Dietterich, and Prasad Tadepalli. Proceedings of International Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014.

- ***Multi-Task Structured Prediction for Entity Analysis: Search-based Learning Algorithms.***

  Chao Ma, Janardhan Rao Doppa, Prasad Tadepalli, Hamed Shahbazh, and Xiaoli Fern. Journal of Machine Learning Research (JMLR), Proceedings Track, Vol 77, 16 pages, 2017.

- ***Randomized Greedy Search for Structured Prediction: Amortized Inference and Learning***

  Chao Ma, F A Rezaur Rahman Chowdhury, Aryan Deshwal, Md Rakibul Islam, Janardhan Rao Doppa, and Dan Roth. Proceedings of International Joint Conference on Artificial Intelligence (IJCAI), 2019.

Thank you!

Questions

Supplementary Material

Structured Prediction with Deep Neural Networks

Papers

- *Structured Prediction Energy Networks (ICML16)*
- *End-to-End Learning for Structured Prediction Energy Networks (ICML17)*
- *Learning Approximate Inference Networks for Structured Prediction (ICRL18)*
- *Deep Structured Prediction with Nonlinear Output Transformations (NIPS18)*

- *Deep Value Networks Learn to Evaluate and Iteratively Refine Structured Outputs (ICML17)*

- *Gradient-based Inference for Networks with Output Constraints (AAAI19)*

Structured Prediction Energy Networks (SPEN)

David Belanger, Andrew McCallum

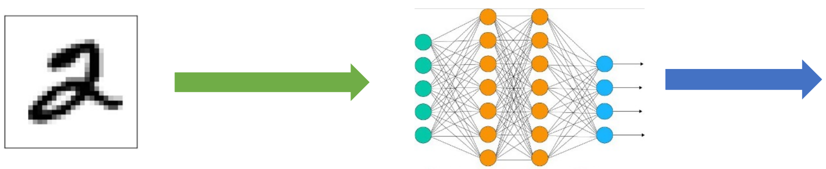
1. Formulating the output as a binary vector $y \in \{0, 1\}^T$, then relax y to real value space so that $\tilde{y} \in [0, 1]^T$.
2. Define an energy network E(x, y) to scoring a given pair of input x and output y.
3. To do inference for a given input x, the best y can be found by compute *argmin*_y *E*(x,y). *argmin* is done by doing gradient descent in the relax output space.
4. Learning of *E* optimizes a SSVM style loss, with inner-loop loss-augmented inference.

Structured Prediction Energy Networks (SPEN)

David Belanger, Andrew McCallum

Formulating the output as a binary vector $y \in \{0, 1\}^T$, then relax y to real value space so that $\tilde{y} \in [0, 1]^T$.

A output of traditional NN for classification:



0	0	1	0	0	0	0	0	0	0

one-hot vectors

e.g., a network of MNIST classification

A output representation in SPEN:

$T = 3$

y = [1, 2, 4]

0	1	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	0

y: concatenation of T one-hot vectors

Structured Prediction Energy Networks (SPEN)

David Belanger, Andrew McCallum

$\hat{y} = \text{argmin}_y\, E(x, y)$. *argmin* is done by doing gradient descent in the relax y space.

Manually:

Define a step_size η and number-of-steps T, and an initial output y_0
At each step, update output by doing following:

$$\mathbf{y}_T = \mathbf{y}_0 - \sum_{t=1}^{T} \eta_t \frac{d}{d\mathbf{y}} E_{\mathbf{x}}(\mathbf{y}_t).$$

Automatically:

Use the GradientDescent optimizer from the library to finish the task above.

Note: After got the real value $\tilde{y}$, a threshold value should be decided through validation set to round it back to discrete space.

Structured Prediction Energy Networks (SPEN)

David Belanger, Andrew McCallum

Energy network learning in SPEN

Loss Function: $$\sum_{\{x_i, y_i\}} \boxed{\max_y \left[\Delta(y_i, y) - E_{x_i}(y)\right.} \left. + E_{x_i}(y_i)\right]_+ .$$

$$[\cdot]_+ = \max(0, \cdot).$$

Loss-augmented inference during training.

$$y_p = \arg\min_y \left(-\Delta(y_i, y) + E_{x_i}(y)\right) .$$

Relax to real value space.

Should work on real value y.

- SPEN relies on a good initial weight. It usually requires a pretraining with a light-weighted network

Structured Prediction Energy Networks (SPEN)

David Belanger, Andrew McCallum

Potential problems of SPEN and gradient based inference

- The outputs during inference are not exploited.
- The gradient based inference requires a lot of parameters.
- Requires pretraining to initialize the network weights.

End-to-End SPEN

David Belanger, Andrew McCallum

Three improvement over the original SPEN:

1. Improve unstable problem of gradient based inference.

Instead, we have found it useful to avoid constrained optimization entirely, by optimizing un-normalized logits $\mathbf{l}_t$, with $\mathbf{y}_t = \text{SoftMax}(\mathbf{l}_t)$:

$$\mathbf{l}_{t+1} = \mathbf{l}_t - \eta_t \nabla E_{\mathbf{x}} \left(\text{SoftMax}(\mathbf{l}_t) \right). \qquad (8)$$

2. Improve the gradient based inference to converge faster.

- Maintain the same optimization configuration, such as T, at both train and test time.
- Exploit all the outputs of each iterate to do the update $L = \frac{1}{T} \sum_{t=1}^{T} w_t \ell(\mathbf{y}_t, \mathbf{y}^*)$, where the w could be $w_t = \frac{1}{T-t+1}$ in practice.
- Set T to a small value.

3. Examples of applying SPEN on different tasks.

Inference Network for SPEN

Lifu Tu, Kevin Gimpel

Instead of gradient based inference, another method to compute $\underset{y}{\mathrm{argmin}}\, E(x, y)$ for SPEN: learning another inference network $A(x)$ that can directly generate an output y.

$$\mathbf{A}_\Psi(\boldsymbol{x}) \approx \underset{\boldsymbol{y} \in \mathcal{Y}_R(\boldsymbol{x})}{\mathrm{argmin}}\, E_\Theta(\boldsymbol{x}, \boldsymbol{y})$$

Learning of Inference network:

$$\hat{\Psi} \leftarrow \underset{\Psi}{\mathrm{argmin}} \sum_{\boldsymbol{x} \in X} E_\Theta(\boldsymbol{x}, \mathbf{A}_\Psi(\boldsymbol{x}))$$

Learning of inference network and energy network:

$$\min_\Theta \max_\Phi \sum_{\langle \boldsymbol{x}_i, \boldsymbol{y}_i \rangle \in \mathcal{D}} [\triangle(\mathbf{A}_\Phi(\boldsymbol{x}_i), \boldsymbol{y}_i) - E_\Theta(\boldsymbol{x}_i, \mathbf{A}_\Phi(\boldsymbol{x}_i)) + E_\Theta(\boldsymbol{x}_i, \boldsymbol{y}_i)]_+$$

Loss-augmented inference network A_Φ can not be directly used as A_Ψ in testing. Needs additional training by initializing A_Φ with A_Ψ weight once E is fixed.

Inference Network for SPEN

Lifu Tu, Kevin Gimpel

Form of inference network $A(x)$

The architecture of A_Ψ will depend on the task.

- For MLC, they use a feed-forward network for A_Ψ with a vector output, treating each dimension as the prediction for a single label.
- For sequence labeling, they use an RNN that returns a vector at each position of *x*. We interpret this vector as a probability distribution over output labels at that position.

There is an analogy here to the discriminator in GANs. The energy function is updated so as to enable it to distinguish “fake” outputs produced by $A_\emptyset$ from real outputs y_i

Deep Value Networks (DVN)

Michael Gygli, Mohammad Norouzi, Anelia Angelova

- Learning a value network $v(x, y)$ to scoring input output pair such that v can imitate the function of $v^* = 1 - l(x, y, y^*)$.
- Inference is done with the similar approach as SPEN: output space gradient descent.
- The learning of v contains two critical step: Generating output for each input, and optimize the cross-entropy loss between v and v^* given all (x, y) pairs in each mini-batch.

Difference between DVN and SPEN:

- For SPEN, the absolute energy value is not important, but for DVN, the absolute value matters.
- DVN is optimizing continuous cross-entropy loss between v and v*, while SPEN is optimizing SSVN style hinge loss.

Deep Value Networks (DVN)

Potential problems of DVN:

1. DVN learns more that what is really needed. The absolute predicted value is unnecessary because during inference you will come across a very small proportion of outputs, rather the whole solution space. For some structured output, coming up a fixed length representation might be difficult.

2. DVN learning performance is very sensitive to what outputs that are generated during training.

3. In practice, the gradient based inference in DVN shows a very critical issue: it has no idea ***where to stop***.

One has to carefully choose the parameters of inference: number of steps, and step_size, threshold for rounding, etc.

Deep Value Networks (DVN)

Potential problems of DVN:

2. In practice, the gradient based inference in DVN shows a very critical issue: it has no idea ***where to stop***.

Results of different number-of-steps with DVN and gradient-base inference:

Train with	Test with	Yeast	Bibtex	Bookmarks
Gradient-20-steps	Gradient-20-steps	63.9	44.5	36.6
Gradient-20-steps	Gradient-10-steps	47.8	34.7	31.2
Gradient-20-steps	Gradient-30-steps	60.2	41.3	31.2
Gradient-20-steps	Gradient-40-steps	57.6	37.9	27.8
Gradient-10-steps	Gradient-10-steps	45.5	37.8	-
Gradient-30-steps	Gradient-30-steps	60.6	42.8	-
Gradient-40-steps	Gradient-40-steps	57.9	37.2	-

Table 2: Gradient-based inference testing performances.

Results of performing discrete greedy search given learned DVN value functions:

Algs.	Yeast	Bibtex	Bookmarks
Search-2-steps	56.3	41.5	34.9
Search-3-steps	59.6	39.5	35.1
Search-4-steps	61.1	35.7	30.4
Search-5-steps	63.4	32.3	27.6
Search-6-steps	62.2	29.1	23.1

Table 3: Multi-label classification F1 performance.

Single Task Structure Prediction

Typical (Single-Task) Structured Prediction:

x → $f: X \longrightarrow Y$ → y

Input | model | Output

Learning

$$f(x, y) = w \cdot \underline{\phi(x, y)}$$

Feature Vector

Inference

$$\hat{y} = \underline{\underset{y}{argmax}\, f(x, y)}$$

Intractable in most cases

Candidate Methods:

- Structural Perceptron
- Structural SVM

......

Candidate Methods:

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

......

This Work

Search Based Inference for MTSP

Complete Output Search Space:

- **State:** y A complete structural output

  e.g.: a document with 5 mentions, y_{ner} = (ORG, PER, PER, LOC, VEL)

- **Action:** $a = (i, v_j, v_k)$ Change the value of i th variable from v_j to v_k

  e.g.: (ORG, $\boxed{\text{PER,}}$ PER, LOC, VEL) $\xrightarrow[\text{Execute action}]{a = (2,\ \text{PER, ORG})}$ (ORG, $\boxed{\text{ORG}}$, PER, LOC, VEL)

- **Successor Function:** $A(y)$ Set of all possible child states of y

  Assume $T = |y|$, and d is domain size, then $|A(y)| = (d-1)T$

- **Initial State:** y_0 Random output or prediction output of unary classifier.

  e.g.: use a ***multi-class classifier*** to predict a label on each mention, and use these predictions as initial output

- **Terminal State:** $\hat{y}$ An output that reaches local optimal cost

  With respect to the outputs in ***beam*** and all ***successor*** outputs

Partial vs. Complete Output Space

Partial vs. *Complete*

Partial

Advantage:

Branching factor is small. Greedy search can be very fast.
Explicit initial and terminal state.

Limitation:

Can be overcome with ***beam***: i.e., LaSO, seq2seq-BSO

- **Requires an ordering.**
- **Value function (ground truth) supervision is only local oracle.**

LSTMs or GRUs with ***Attention***

Complete

Advantage:

All the disadvantages of Partial space can be overcome.

Limitation:

- **Requires initial states, and not explicit terminal state.**
- **Computational expensive on both time and space.**