Mining Entity Synonyms with Efficient Neural Set Generation

Jiaming Shen[1]**,** Ruiliang Lyu[2], Xiang Ren[3], Michelle Vanni[4], Brian Sadler[4], Jiawei Han[1]

[1]University of Illinois at Urbana-Champaign [2]Shanghai Jiao Tong University

[3]University of Southern California [4]U.S. Army Research Laboratory

Presented by Jiaming @ AAAI 2019

Code: https://github.com/mickeystroller/SynSetMine-pytorch

Data: http://bit.ly/SynSetMine-dataset

Outline

- Introduction
- Related Work
- Problem Formulation
- Our Proposed Framework
- Experiments
- Summary

Introduction

- An **entity synonym set** is a set of terms (*i.e.*, words or phrases) referring to the same entity:
  - Singular/Plural Derivation
  - Acronym/Abbreviation
  - Slogan/Slang
  - ……

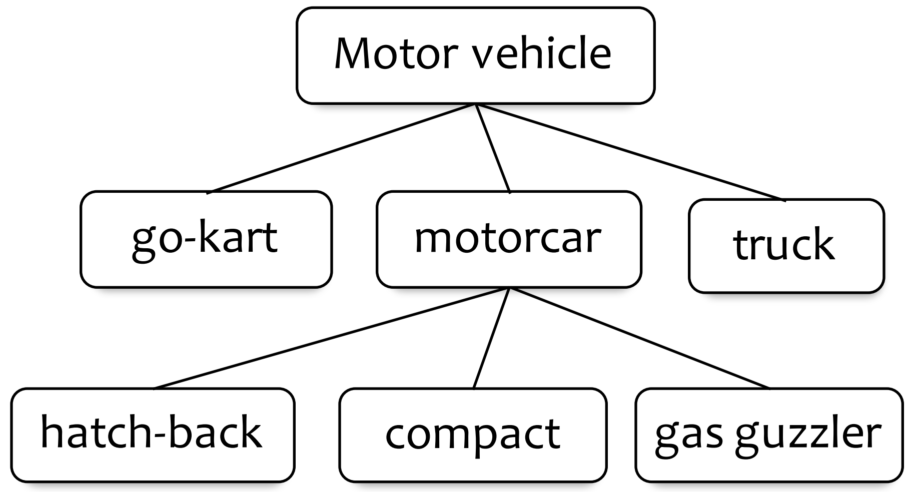
Entity Synonym Set
{United States, U.S., USA, the US}
{United Arab Emirates, UAE, the Emirates}
{Emirates Airline, Emirates Air}

- Mining entity synonym sets can benefit lots of **entity-leveraging applications**:
  - Query Understanding & Web Search
  - Question Answering & Dialog System
  - Taxonomy Construction & Enrichment
  - ……

Query	*Emirates Airline U.S. to UAE*

Knowledge Web

Intelligent Assistant



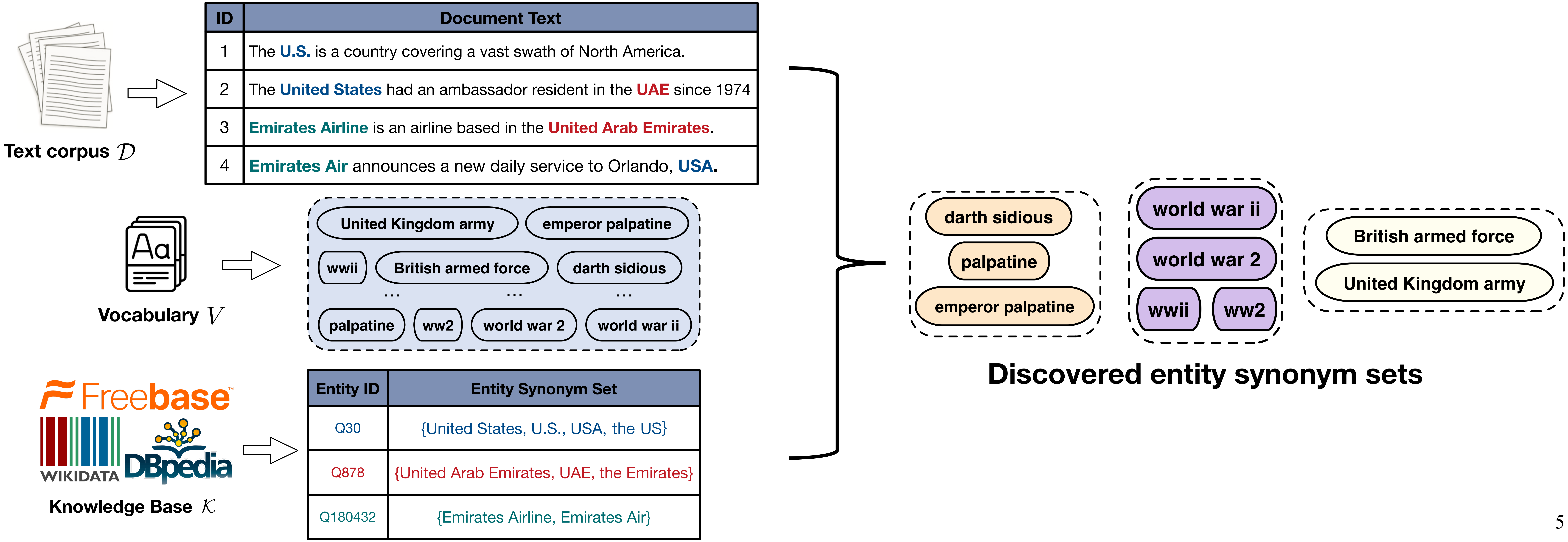
Taxonomy Construction

Related Work

- **Approach 1: Ranking + Pruning**
  - Given a query term, first rank all candidate terms, then prune the rank list into an output set
  - For example, “United States” -> [“U.S.”, “U.S.A”, “UAE”, …,] -> {“U.S.”, “U.S.A”}
  - Pros: 1) leverage heterogeneous training signals; 2) suitable for online query-dependent applications
  - Cons: 1) ignore the relation between candidate terms; 2) non-trivial to convert rank list to set output
- **Approach 2: Synonymy Detection + Organization**
  - Given a vocabulary, first find all synonym pairs, then aggregation these pairs into synonym sets
  - For example, [“U.S.”, “U.S.A”, “UAE”, “the US”, “the Emirates”, …] -> [(“U.S.”, “U.S.A”), (“U.S.”, “the US”), (“UAE”, “the Emirates”)] -> {{“U.S.”, “U.S.A”, “the US”}, {“UAE”, “the Emirates”}}
  - Pros: 1) model the candidate term relations; 2) return all synonym sets in vocabulary
  - Cons: 1) cannot leverage signals for synonym organization; 2) error propagation between two phases

Our Problem Formulation

- **Input**: (1) a text corpus *D*, (2) a vocabulary *V* (i.e., a list of terms) derived from *D*, and (3) a knowledge base *K* (consists of known entity synonym sets)
- **Output**: All entity synonym sets consisting of terms in *V*



ID	Document Text
1	The **U.S.** is a country covering a vast swath of North America.
2	The **United States** had an ambassador resident in the **UAE** since 1974
3	**Emirates Airline** is an airline based in the **United Arab Emirates**.
4	**Emirates Air** announces a new daily service to Orlando, **USA**.

Entity ID	Entity Synonym Set
Q30	{United States, U.S., USA, the US}
Q878	{United Arab Emirates, UAE, the Emirates}
Q180432	{Emirates Airline, Emirates Air}

Our Proposed Framework: **SynSetMine**

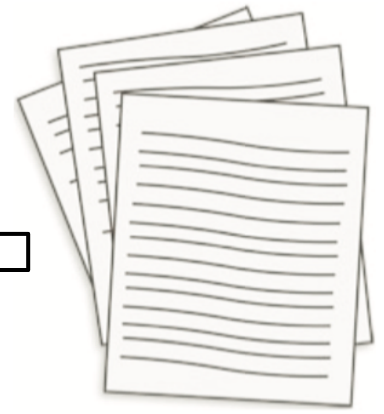
- **SynSetMine** framework consists of three major steps:
  - Step 1: Acquire Distant Supervision by entity linking
  - Step 2: Learn Set-Instance Classifier using distant supervision
  - Step 3: Apply Set Generation Algorithm based on learned classifier

Step 1: Distant Supervision Acquisition

- Use entity linkers to map in-corpus text (*i.e.*, entity mentions) to entities in knowledge base
- Group all entity mentions that linked to the same entity as a training entity synonym set

Distant Supervision Acquisition

ID	Document Text
1	The **U.S.** is a country covering a vast swath of North America.
2	The **United States** had an ambassador resident in the **UAE** since 1974
3	**Emirates Airline** is an airline based in the **United Arab Emirates**.
4	**Emirates Air** announces a new daily service to Orlando, **USA.**



Text corpus $\mathcal{D}$

Entity linking

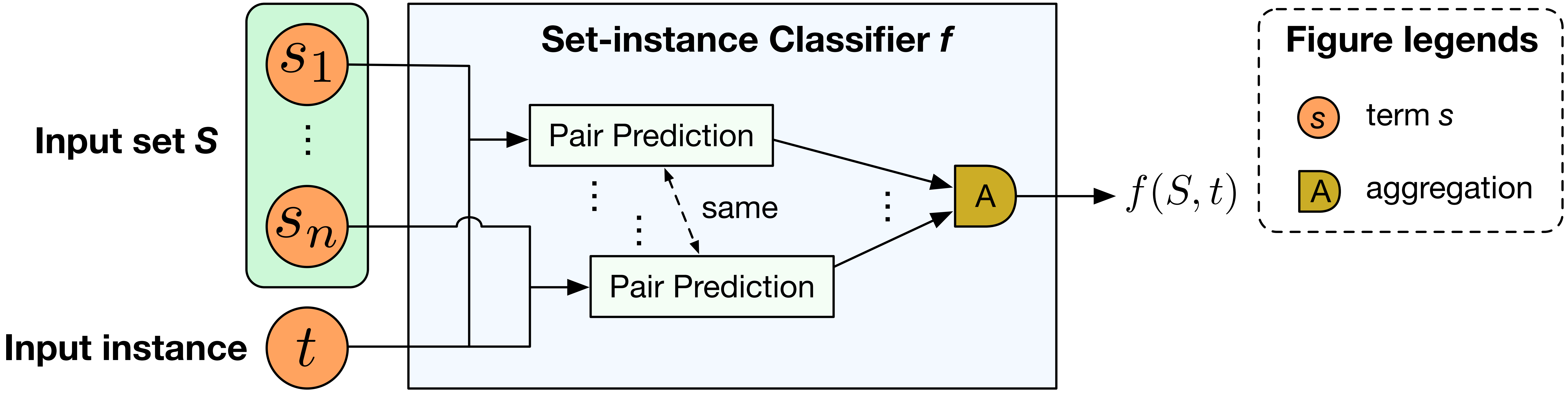
Knowledge Base $\mathcal{K}$

Distant Supervision

Entity ID	Entity Synonym Set
Q30	{United States, U.S., USA, the US}
Q878	{United Arab Emirates, UAE, the Emirates}
Q180432	{Emirates Airline, Emirates Air}

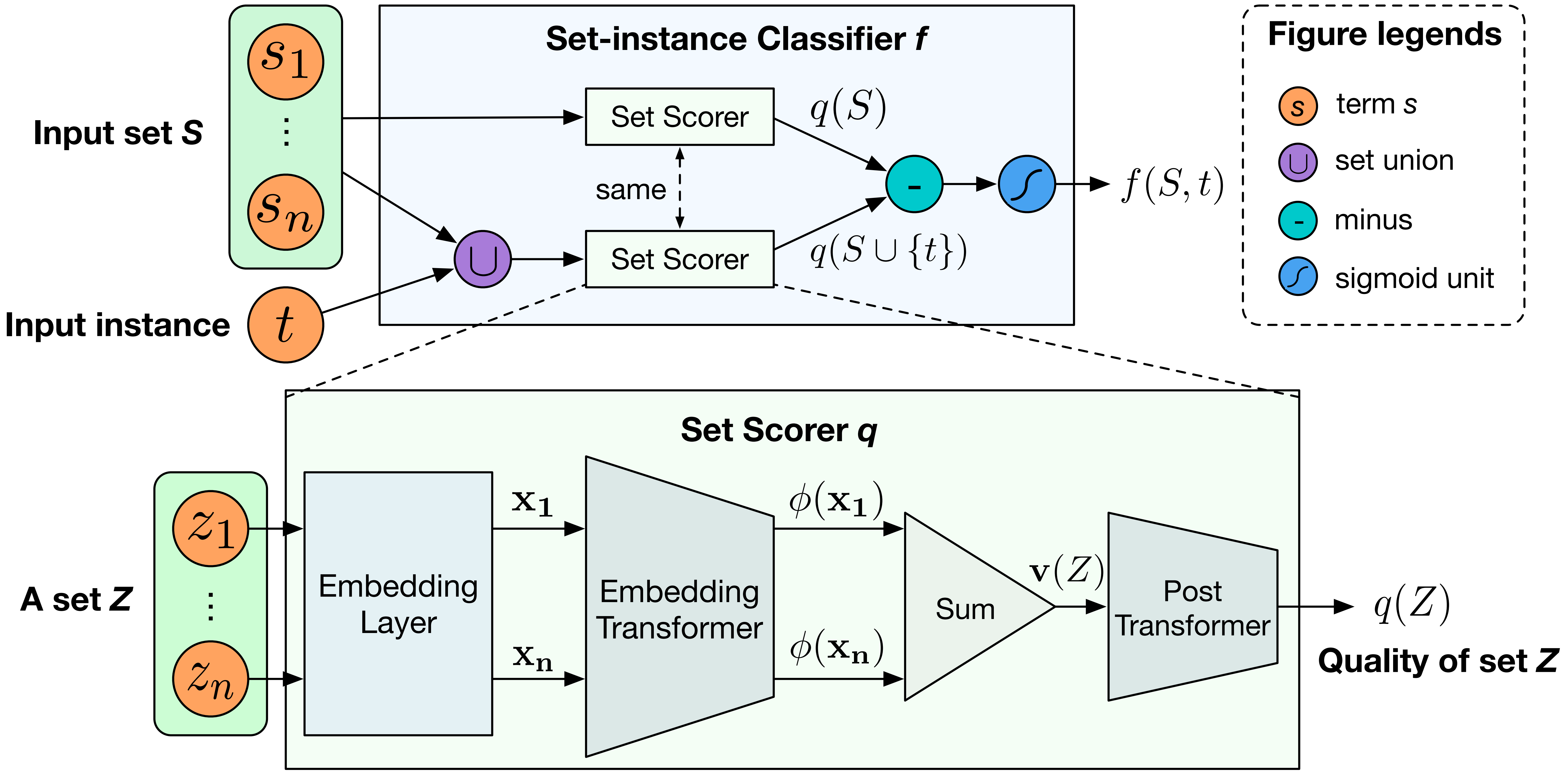
Step 2: Set-Instance Classifier Architecture

- A **set-instance classifier** *f(S, t)* returns the probability that an instance *t* (*i.e.*, a term) belongs to an entity synonym set *S*
  - Need to be **invariant to the ordering of elements in set *S***
  - For example, if *f({"USA", "the U.S."}, "U.S.")* is 0.9, then *f({"the U.S.", "USA"}, "U.S.")* should also be 0.9
- An intuitive way to achieve this goal is **aggregate pair prediction** results:
  - However, this approach fails to model the holistic set semantics



Step 2: Set-Instance Classifier Architecture (Cont'd)

- Our approach in SynSetMine:
  - Use set representation learning to construct a **set scorer** which outputs the quality score of a set
  - Construct the set-instance classifier using the set scorer



Step 2: Learning Set-Instance Classifier

- To learn the set-instance classifier, we need to convert the distant supervision (in the form of entity sets) to a collection of set-instance pairs, using different negative sampling strategies:
  - **Completely random**: ({“Emirates Air”, “Emirates Airline”}, “the US”, 0)
  - **Share token**: ({“United Arab Emirates”, “UAE”}, “Emirates Air”, 0)
  - **Mixture** of above two strategies

Entity ID	Entity Synonym Set
Q30	{United States, U.S., USA, the US}
Q878	{United Arab Emirates, UAE, the Emirates}
Q180432	{Emirates Airline, Emirates Air}

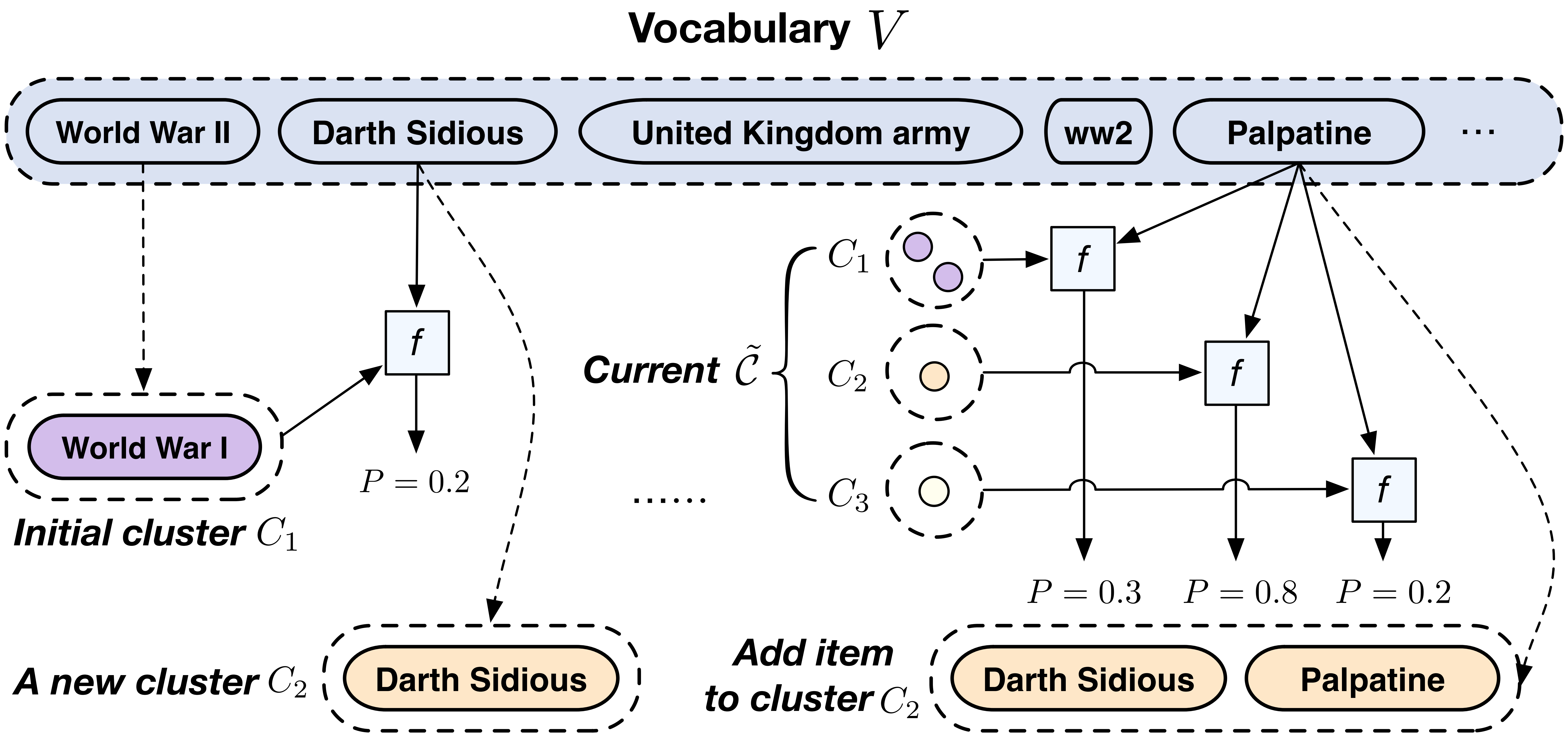
Original Distant Supervision Format

Set	Instance	Label
{United States, U.S.}	USA	1
{United States, USA.}	UAE	0
{United Arab Emirates, UAE}	the Emirates	1
{United Arab Emirates, UAE}	Emirates Airline	0
{Emirates Airline}	Emirates Air	1
{Emirates Air}	the US	0

Converted Format for Training Set-Instance Classifier

Step 3: Apply Set Generation Algorithm

- Apply learned set-instance classifier to extract new synonym sets in the vocabulary
  - Enumerate the vocabulary once and output all synonym sets in the vocabulary
  - Determine for each term whether it can be put into an existing cluster or create a new singleton cluster



Leverage learned set-instance classifier to make the decision

Step 3: Apply Set Generation Algorithm (Cont'd)

- Apply learned set-instance classifier to extract new synonym sets in the vocabulary

Algorithm 1: Set Generation Algorithm

Input: A set-instance classifier f; An input vocabulary $V = (s_1, s_2, \ldots, s_{|V|})$; A threshold $\theta \in [0, 1]$.

Output: m entity synonym sets $\mathcal{C} = [C_1, C_2, \ldots, C_m]$ where $C_i \subseteq V, \cup_{i=1}^{m} C_i = V, C_i \cap C_j = \emptyset, \forall i \neq j$.

$\mathcal{C} \leftarrow [\{s_1\}]$; // initialize the first single-element cluster;
for i *from 2 to* $|V|$ **do**
    $best\_score = 0$;
    $best\_j = 1$;
    for j *from 1 to* $|\mathcal{C}|$ **do**
        if $f(C_j, s_i) > best\_score$ **then**
            $best\_score \leftarrow f(C_j, s_i)$;
            $best\_j \leftarrow j$;
    if $best\_score > \theta$ **then**
        $C_{best\_j}.add(s_i)$;
    else
        $\mathcal{C}$.append($\{s_i\}$); //add a new cluster into the output;
Return $\mathcal{C}$;

Find best matching cluster

Either add term into best matching cluster
or create a new singleton cluster

Experimental Setups

- **Datasets**:
  - For Wiki/NYT datasets, using DBpedia Spotlight as entity Linker
  - For PubMed dataset, using PubTator as entity Linker
  - Available for download: http://bit.ly/SynSetMine-dataset
- **Evaluation metrics**:
  - Adjusted Rand Index (ARI)
  - Fowlkes-Mallows Index (FMI)
  - Normalized Mutual Information (NMI)

Table 1: Datasets Statistics.

Dataset	**Wiki**	**NYT**	**PubMed**
#Documents	100,000	118,664	1,554,433
#Sentences	6,839,331	3,002,123	15,051,203
#Terms in *train*	8,731	2,600	72,627
#Synonym sets in *train*	4,359	1,273	28,600
#Terms in *test*	891	389	1,743
#Synonym sets in *test*	256	117	250

Experimental Setups (Cont'd)

- **Compared methods:**
  - Kmeans: an unsupervised feature-based clustering algorithm
  - Louvain: an unsupervised community detection algorithm
  - SetExpan+Louvain: use SetExpan (a set expansion algorithm) to construct graph, then apply Louvain
  - COP-Kmeans: a semi-supervised clustering algorithm
  - SVM+Louvain: supervised pair prediction for graph construction and then apply Louvain
  - L2C: supervised learning to cluster algorithm
  - SynSetMine: Our proposed approach
- Our model Implementation: https://github.com/mickeystroller/SynSetMine-pytorch

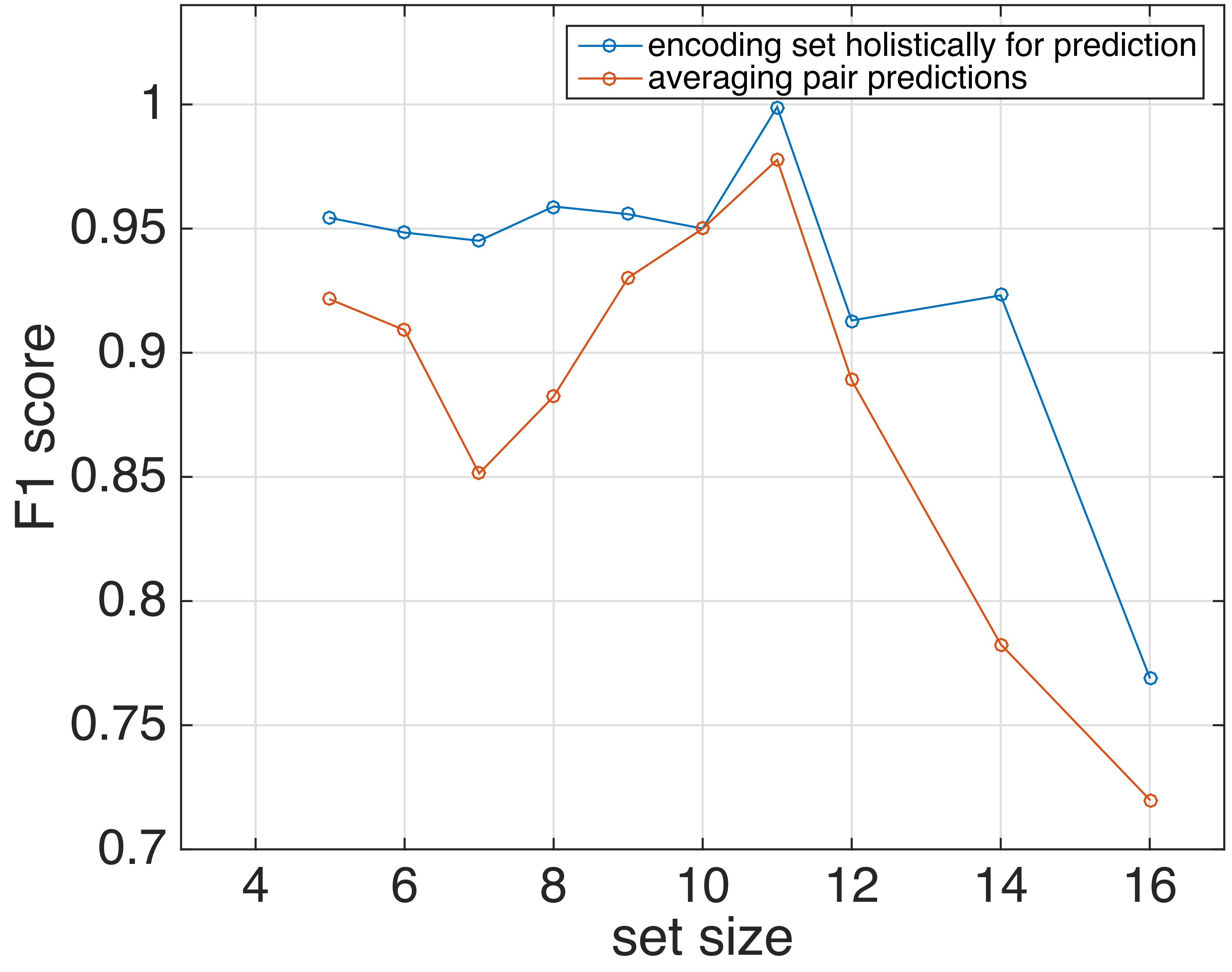
Experimental Results – Clustering Performance

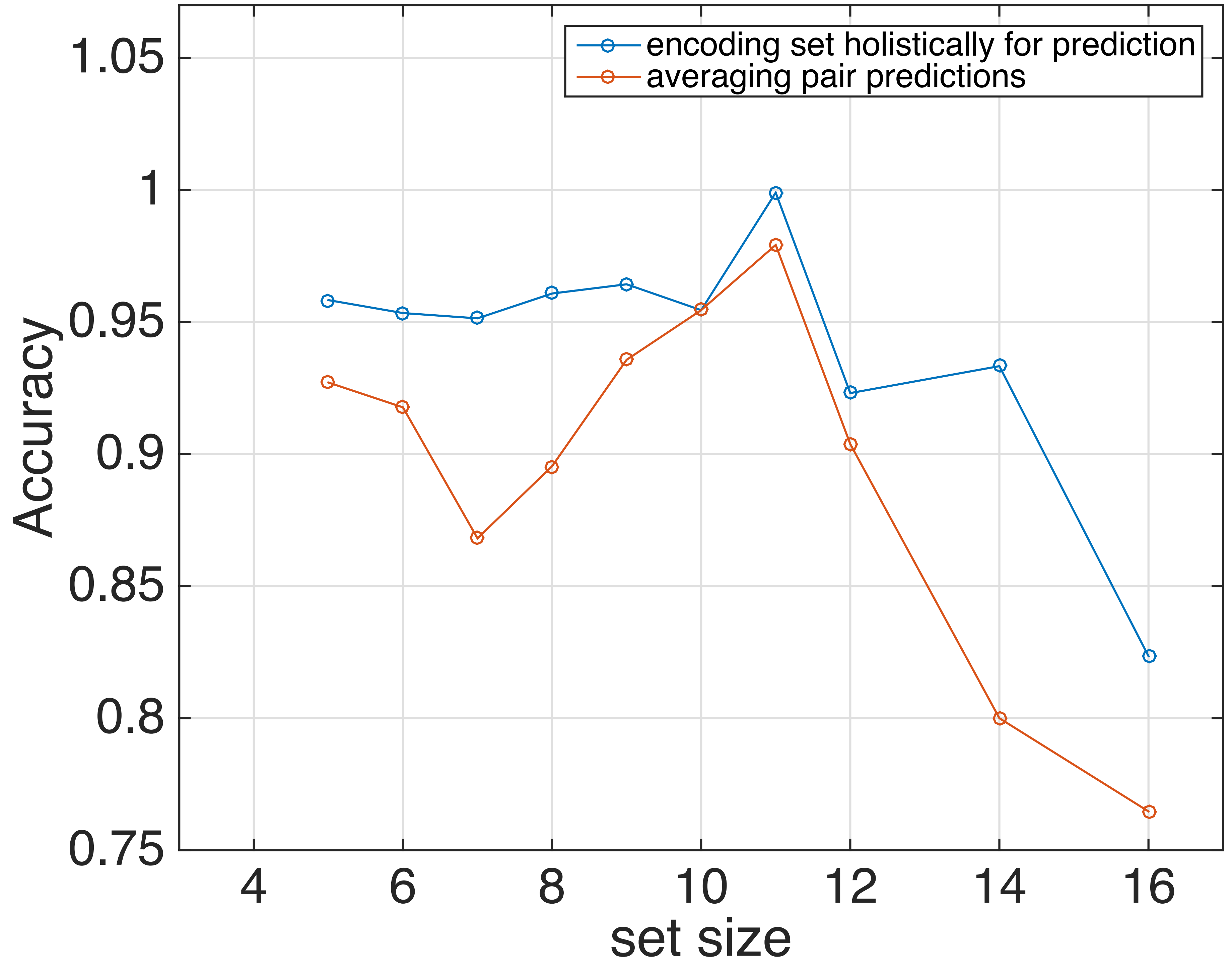
- Overall clustering performance:
  - SetExpan+Louvain > Louvain: use SetExpan for graph construction is useful
  - COP-Kmeans > Kmeans: additional supervision signals is useful
  - SVM+Louvain & L2C's bad performance: effectively use supervision signals is challenging
  - SynSetMine > all: the effectiveness of our proposed approach

Method	Wiki			NYT			PubMed		
	ARI (±std)	FMI (±std)	NMI (±std)	ARI (±std)	FMI (±std)	NMI (±std)	ARI (±std)	FMI (±std)	NMI (±std)
Kmeans	34.35 (±1.06)	35.47 (±0.96)	86.98 (±0.27)	28.87 (±1.98)	30.85 (±1.76)	83.71 (±0.57)	48.68 (±1.93)	49.86 (±1.79)	88.08 (±0.45)
Louvain	42.25 (±0.00)	46.48 (±0.00)	92.58 (±0.00)	21.83 (±0.00)	30.58 (±0.00)	90.13 (±0.00)	46.58 (±0.00)	52.76 (±0.00)	90.46 (±0.00)
SetExpan+Louvain	44.78 (±0.28)	44.95 (±0.28)	92.12 (±0.02)	43.92 (±0.90)	44.31 (±0.93)	90.34 (±0.11)	58.91 (±0.08)	61.87(±0.07)	92.23 (±0.15)
COP-Kmeans	38.80 (±0.51)	39.96 (±0.49)	90.31 (±0.15)	33.80 (±1.94)	34.57 (±2.06)	87.92 (±0.30)	49.12 (±0.85)	51.92 (±0.83)	89.91 (±0.15)
SVM+Louvain	6.03 (±0.73)	7.75 (±0.81)	25.43 (±0.13)	3.64 (±0.42)	5.10 (±0.39)	21.02 (±0.27)	7.76 (±0.96)	8.79 (±1.03)	31.08 (±0.34)
L2C	12.87 (±0.22)	19.90 (±0.24)	73.47 (±0.29)	12.71 (±0.89)	16.66 (±0.68)	70.23 (±1.20)	–	–	–
SynSetMine	**56.43 (±1.31)**	**57.10 (±1.17)**	**93.04 (±0.23)**	**44.91 (±2.16)**	**46.37 (±1.92)**	**90.62 (±1.53)**	**74.33 (±0.66)**	**74.45 (±0.64)**	**94.90 (±0.97)**

Experimental Results – Set-Instance Pair Prediction

- Effectiveness of using set representation learning for set-instance classifier:
  - Tested on 3486 set-instance pairs, half of them are positive pairs
  - Use F1 score and Accuracy for evaluation





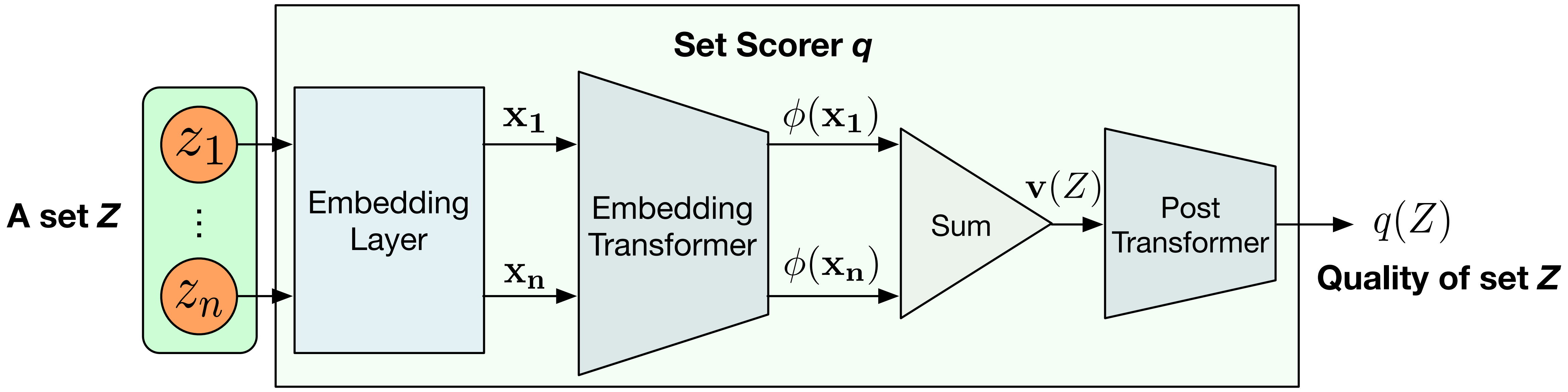
Experimental Results – Efficiency Analysis

- Efficiency of set generation algorithm
  - SynSetMine is faster to train compared with the other neural network based method L2C
  - SynSetMine is the fastest during prediction stage (which returns all synonym sets in vocabulary)

Method	Training			Prediction		
	Wiki	NYT	PubMed	Wiki	NYT	PubMed
Kmeans	–	–	–	1.82s	0.88s	2.95s
Louvain	–	–	–	3.94s	20.59s	74.6s
SetExpan+Louvain	–	–	–	323s	120s	4143s
COP-KMeans	–	–	–	249s	37.94s	713s
SVM+Louvain	4.9m	37s	1.3h	29.21s	5.80s	101.32s
L2C	16.8h⋆	30.7m⋆	>120h⋆	20.9m⋆	56.6s⋆	–
SynSetMine	48m⋆	6.5m⋆	7.5h⋆	0.852s⋆	0.348s⋆	1.84s⋆

Experimental Results – Model Architecture Analysis

- Importance of different model components:
  - Embedding Layer: Pre-trained 50-d word embedding
  - Embedding Transformer (ET): A two-layer NN of sizes {50, X}
  - Post Transformer (PT): A three-layer NN of sizes {X, Y, X}



Method	Wiki			NYT		
	ARI	FMI	NMI	ARI	FMI	NMI
No-ET	46.48	47.23	91.57	39.86	42.67	90.46
No-PT	1.50	0.50	89.95	0.82	1.70	82.20
Both-100-200	49.38	49.56	91.21	37.64	39.37	89.33
Both-150-300	53.06	53.27	91.96	43.20	44.08	89.57
Both-200-400	53.82	53.99	92.36	47.03	49.65	91.00
Both-250-500	**57.34**	**58.13**	**93.10**	**48.89**	**51.33**	**91.19**
Both-300-600	56.26	56.51	92.92	46.65	47.30	90.01
Both-350-600	55.93	56.10	92.69	47.40	48.37	90.14

Two transformers are both Important

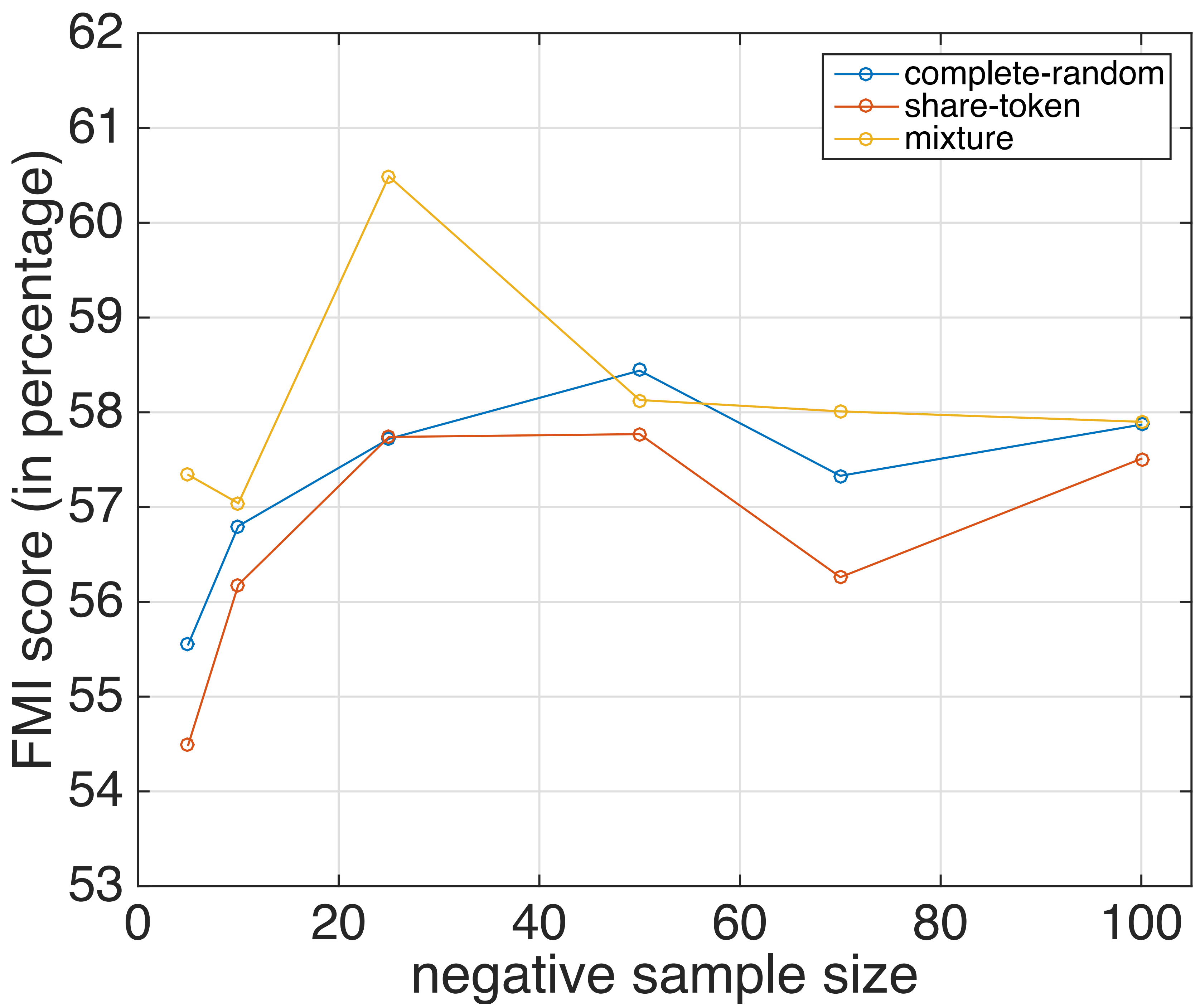
Recommended Architecture

Experimental Results – Negative Sampling Analysis

- Effect of negative set-instance pair sampling methods:
  - Complete-random works surprisingly well
  - Mixture of shared-token and complete-random schemes are the best
  - The “diversity” of negative examples is important

Entity ID	Entity Synonym Set
Q30	{United States, U.S., USA, the US}
Q878	{United Arab Emirates, UAE, the Emirates}
Q180432	{Emirates Airline, Emirates Air}

Set	Instance	Label
{United States, U.S.}	USA	1
{United States, USA.}	UAE	0
{United Arab Emirates, UAE}	the Emirates	1
{United Arab Emirates, UAE}	Emirates Airline	0
{Emirates Airline}	Emirates Air	1
{Emirates Air}	the US	0



Experimental Results – Case Studies

- Example outputs on three datasets:

Dataset	Distant Supervision	Discovered Synonym Sets
Wiki	{"*londres*", "*london*"}	{"*gas*", "*gasoline*", "*petrol*"}
	{"*mushroom*", "*toadstool*"}	{"*roman fort*", "*castra*"}
NYT	{"*myanmar*", "*burma*"}	{ "*royal dutch shell plc*", "*royal dutch shell*", "*shell*" }
	{"*honda motor*", "*honda*"}	{"*chief executive officier*", "*ceo*" }
PubMed	{"*alzheimers disease*", "*Alzheimer's dementia*"}	{"*dna microarrays*", "*dna chip*", "*gene expression array*", "*dna array*"}

- Comparison of set-instance classifier with the approach that aggregates instance-instance pair prediction results:

Method	Set-instance Classifier	Aggregate Pair Predictions
Synonym set	{"*u.k.*", "*britain*"}	{"*u.k.*", "*britain*"}
Ranked terms	"*uk*" "*united kingdom*" "*great britain*" "*elizabeth ii*"	"*uk*" "*indie*" "*united kingdom*" "*america*"

Summary

- Synonym Set Discovery Task:
  - Given a corpus D, a knowledge base K, and a vocabulary V, output all entity synonym sets in V
- SynSetMine Framework:
  - Leverage Knowledge Base to obtain distant supervision
  - Learn an accurate set-instance classifier
  - Integrate the set-instance classifier into an efficient set generation algorithm
- Conclusions:
  - Modeling a set holistically is important
  - Generating “diverse” set-instance pairs for training set-instance classifier is important

Future Work

- Extend SynSetMine to weakly-supervised setting:
  - Users provide a small set of “seed” synonym sets for model learning
- Extend SynSetMine’s philosophy to other set prediction and clustering tasks:
  - Supervised Clustering
  - Metric Learning
- Further integrate set-instance classifier into the set generation algorithm and learn both of them in an end-to-end fashion

Thanks

Questions?