# Incorporating Term Definitions for Taxonomic Relation Identification

Yongpan Sheng[1] Tianxing Wu[2] Xin Wang[3]

[1]School of Computer Science and Technology
University of Electronic Science and Technology of China (UESTC),
[2]Nangyang Technological University, [3]Tianjin University

The 9th Joint International Semantic Technology Conference (JIST), 2019

# Outline

# Outline

# Motivation

People often organize the lexical knowledge in the form of term taxonomy, such as Cognitive Concept Graph, HowNet.

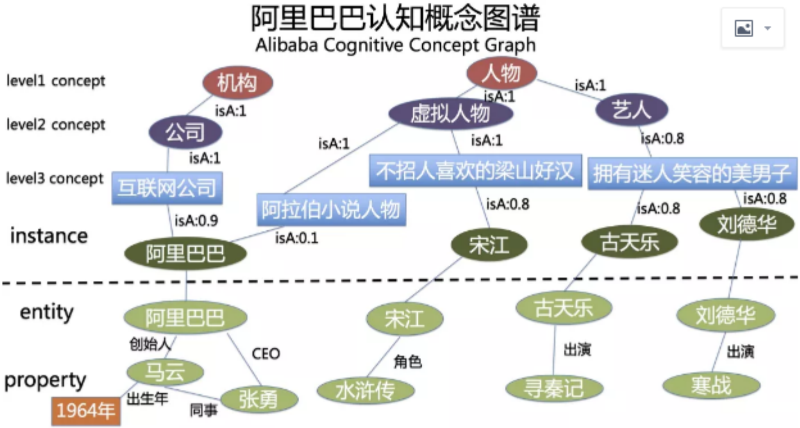



Figure: An example of Cognitive Concept Graph from Alibaba

# Motivation

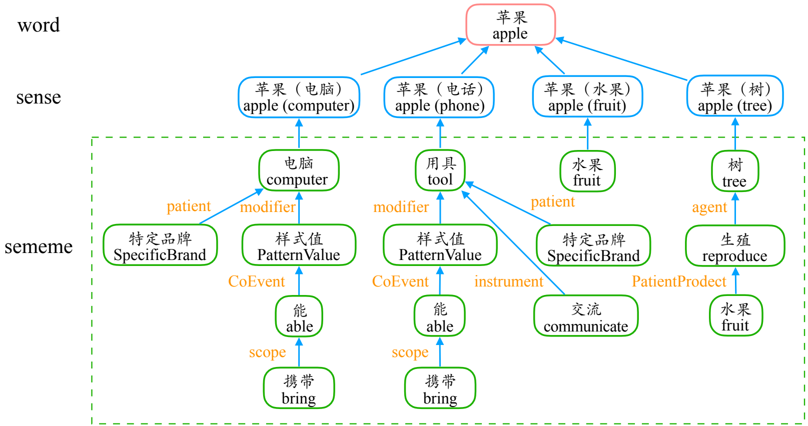
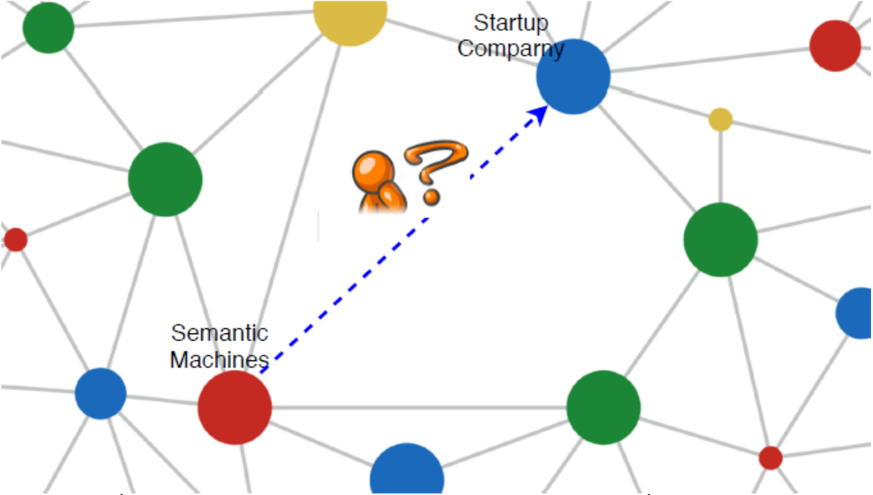



Figure: An example of word annotated with sememes in HowNet

# Motivation




Predicting (*Semantic Machines*, *isA*, *Startup Company*) mainly contributes to knowledge graph completion.

# Outline

# Problem Definition

Determine whether a specific pair of terms [1] holds the taxonomic relation ("isA" relation) or not.

e.g.,

- ("*Einstein*", "*scientist*")
- ("*Mel Gibson*", "*actor*")
- ("*Paris*", "*city*")

[1] "terms" refers to any words or phrases.

# Outline

# Previous approaches

**Linguistic approaches**, mainly rely on lexical-syntactic patterns (e.g., A typical pattern is "A such as B")

- Higher precision in several applications, e.g., Probase construction.

The main drawbacks of this type of method are:

- Identified patterns are too specific to cover the wide range of complex linguistic circumstance.
- Fully unsupervised, e.g., they may require a set of seed instances to initiate the extraction process. Hence, recall is sacrificed.

# Previous approaches

**Distributional approaches**, embed the two terms into context-aware vector representations, and then predict their taxonomic relation based on these representations.

The main drawbacks of this type of method are:

- Domain specificity
  An IT corpus hardly mentions "apple" as a fruit.
- Poor generalization capability
  (i) **Other taxonomic relations**.
  E.g., distributional inclusion hypothesis[2] - if ("*Einstein*", "*scientist*"), the typical contexts of *Einstein* will occur also with *scientist*.
  (ii) **Unseen terms**, **rare terms**, and **terms with biased sense distribution**.
  E.g., *Unseen terms* - word embeddings for specified taxonomic relation[3].

[2] Harris, Z.S., Distributional Structure, 1954.

[3] Nguyen, Kim Anh and Köper, Maximilian and Walde, et al., Hierarchical embeddings for hypernymy detection and directionality, EMNLP 2017.

# Previous approaches

- Poor generalization capability
  (ii) *Unseen terms*, *rare terms*, and *terms with biased sense distribution*.
  E.g., *rare terms* - (“*coma*”, “*knowledge*”), (“*bacterium*”, “*microorganism*”)
  *terms with biased sense distribution* -
  (“*apple*”, “*fruit*”),
  (“*apple*”, “*IT company*”)

# Previous approaches

Limitations of the previous methods.

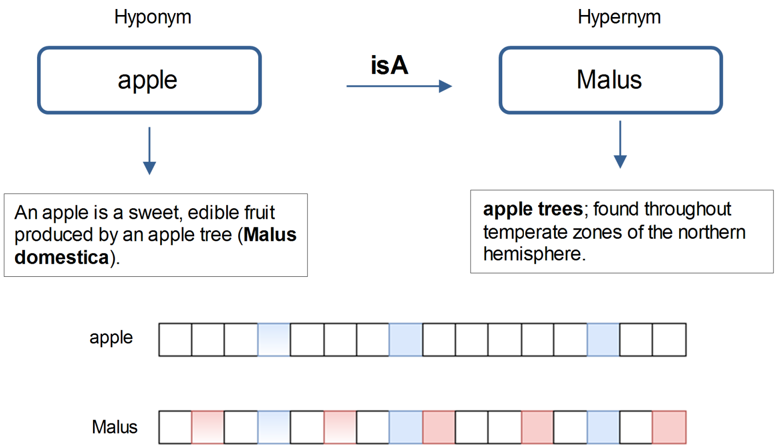
- Lower recall (Linguistic approach)
- Domain specificity (Distributional approach)
- Poor generalization capability (Distributional approach)

As a result, the performance of this task is **far from satisfactory**.

# Outline

# Outline

# Our Inspirations



# Our Inspirations

Richer interpretation (**definition in sense-level**, **distributional context**)
Our model is expected to:

- Accurate prediction of taxonomic relations of term pairs in sense-level.
- Generalizing well to unseen terms, rare terms, and terms with biased sense distribution.

# Outline

# The Baseline System

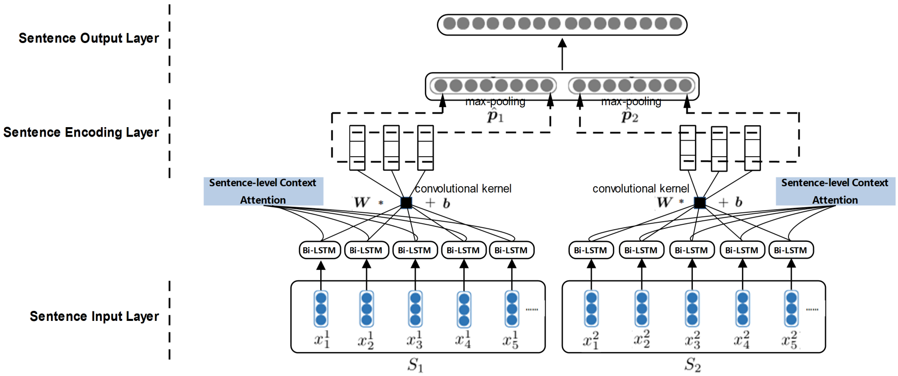



Figure: Architecture of the baseline system

# Our Proposed Model

**Sentence Encoding Layer.**

- Idea: Siamese Network[4]
- Architecture: **Bi-LSTM + Sentence-level Context Attention + CNN Input**: sentence pair $S_1$ and $S_2$. In our settings, term can be treated as short sentence.
  **Output**: the neural representation $\hat{p}_i$ of each sentence $S_i$ ($i = 1, 2$)

**Sentence Output Layer.**
The overall representation for the sentence pair, i.e., concatenating $\hat{p}_1$ and $\hat{p}_2$.

[4] Neculoiu, P., Versteegh, M., et al., Learning text similarity with siamese recurrent networks, the 1st Workshop on Representation Learning for NLP 2016

# Outline

# Our Proposed Model

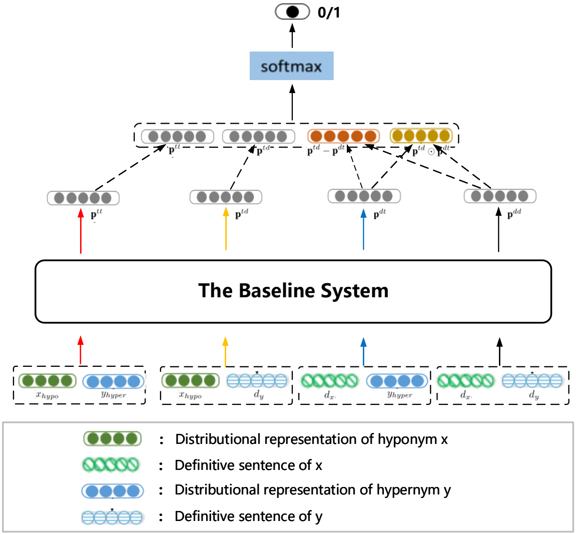



Figure: Architecture of our proposed model

# Our Proposed Model

**The Sentence Input Strategy.** Four strategies to define the input representations on the baseline system:

- $\mathbf{p}^{tt}$ from $(x_{hypo},\ y_{hyper})$
  This combination intends to model embeddings from a hyponym to its hypernym via a network with weights.
- $\mathbf{p}^{td}$ from $(x_{hypo},\ d_y)$, $\mathbf{p}^{dt}$ from $(d_x,\ y_{hyper})$
  These combinations benefit to generate indicative features across distributional context and definition for discriminating taxonomic relations from other semantic relations.
- $\mathbf{p}^{dd}$ from $(d_x,\ d_y)$
  This combination provides an alternative evidence for interpreting the terms.

**Heuristic Matching.**

$$\mathbf{p} = [\mathbf{p}^{tt}; \mathbf{p}^{dd}; \mathbf{p}^{td} - \mathbf{p}^{dt}; \mathbf{p}^{td} \odot \mathbf{p}^{dt}], \tag{1}$$

# Our Proposed Model

**Softmax Output.**

$$\mathbf{o} = \mathbf{W}_1(\mathbf{p} \circ \mathbf{r}) + \mathbf{b}. \tag{2}$$

**Loss Function and Training.**

$$p(y_i|x_i, \theta) = \frac{e^{o^i}}{\Sigma_k e^{o_k}}, \tag{3}$$

$$J(\theta) = \sum_i log\ p(y_i|x_i, \theta), \tag{4}$$

**Noting that:**

Given an input instance as ($x_{hyper}$, $d_x$, $y_{hypo}$, $d_y$, 1/0), the network with parameter $\theta$ outputs the vector **o**, which is a 2-dimensional vector with the sum of component probability to 1.
To compute $\theta$, we maximize the log likelihood $J(\theta)$ through stochastic gradient descent over shuffled mini-batches with the Adam update rule.

# Outline

# Outline

## Dataset

Table: Dataset used in the experiments

| Dataset | #Train | | #Test | | #Validation | |
|---|---|---|---|---|---|---|
| Splits | random splits | lexical splits | random splits | lexical splits | random splits | lexical splits |
| **BLESS** [a] | 12459 | 757 | 2376 | 675 | 404 | 103 |
| **Conceptual Graph** [b] | 58484 | 29475 | 19610 | 7808 | 4079 | 2095 |
| **WebIsA-Animal** [c] | 5614 | 3784 | 1942 | 1021 | 407 | 249 |
| **WebIsA-Plant** [c] | 5534 | 2933 | 1610 | 861 | 305 | 169 |

- **Random and Lexical Dataset Splits**. To better address "lexical memorization" phenomenon[d].
  Roughly a ratio of **14:5:1** for training set, test set and validation set partitioned randomly.
  Roughly a ratio of **8:1** for positive instances and negative instances in random or lexical splits in the datasets.

[a] https://sites.google.com/site/geometricalmodels/shared-evaluation
[b] https://concept.research.microsoft.com/Home/Download
[c] http://webdatacommons.org/isadb/

# Experiments and Analysis

## Dataset

- **Term Definition Collection** - WordNet and English Wikipedia.
  (i) We first try to extract respective definition of hyponym and hypernym from **WordNet** based on the term in strings. For a few pairs which contain terms not covered by WordNet. (ii) We then switch to Wikipedia in which the term can be involved, and select the **top-2 sentences** in the first subgraph in the introductory sections, as its definitive description. (iii) If we are failed in two knowledge resources, we set definitions the same as the term in strings.
- **Pre-trained Word Embeddings**. We use two large-scale textual corpus (i.e., English wikipedia and extracted abstracts of DBpedia) to train word embeddings.

## Noting that:

As WordNet sorts sense definitions by sense frequency, we only choose the top-1 sense definition to denote a term.

# Outline

# Experimental settings

## Compared methods:

- **Word2Vec[a] + SVM**
- **DDM[b] + SVM**
- **DWNN[c] + SVM**
- **Best unsupervised** (denpendency-based context)[d]
- **Ours**$_{SubInput}$, the variant of our method
- **Ours**$_{Concat}$, the variant of our method

[a] Mikolov, T., Chen, K., et al., Efficient estimation of word representations in vector space, ICLR (Workshop Poster) 2013

[b] Yu, Z., Wang, H., et al., Learning Term Embeddings for Hypernymy Identification, IJCAI 2015

[c] Anh, T.L., Tay, Y., et al., Learning Term Embeddings for Taxonomic Relation Identification Using Dynamic Weighting Neural Network, EMNLP 2016

[d] Shwartz, V., Santus, E., et al., Hypernyms under Siege: Linguistically-motivated Artillery for Hypernymy Detection, EACL 2017

# Experimental settings

**Evaluation metrics**: Mean Average F-score (for random splits), Average F-score@200 (for lexical splits)

**Parameter Tuning**: We employ grid search for a range of hyper-parameters, and picked the combination of ones that yield the highest F-score on the validation set.

# Outline

## Performance on specific domain datasets

Table: Performance comparison of different methods on domain-specific datasets (including WebIsA-Animal and WebIsA-Plant). We report Precision at rank 200 (P@200), Recall at rank 200 (R@200), F-score at rank 200 (F@200) for Random Splits, and Mean Average Precision (P), Mean Average Recall (R), Mean Average F-score (F) for Lexical Splits. The best performance in the F-score column is boldfaced (the higher, the better).

| Datasets | WebIsA-Animal | | | | | | WebIsA-Plant | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Random Splits (@200) | | | Lexical Splits | | | Random Splits (@200) | | | Lexical Splits | | |
| Method | P | R | F | P | R | F | P | R | F | P | R | F |
| Previous methods | | | | | | | | | | | | |
| **Word2Vec + SVM** | 0.796 | 0.706 | 0.748 | 0.785 | 0.674 | 0.725 | 0.817 | 0.730 | 0.771 | 0.708 | 0.623 | 0.663 |
| **DDM + SVM** | 0.706 | 0.620 | 0.660 | 0.655 | 0.521 | 0.580 | 0.759 | 0.695 | 0.726 | 0.712 | 0.517 | 0.599 |
| **DWNN + SVM** | 0.893 | 0.714 | 0.794 | 0.820 | 0.550 | 0.658 | 0.916 | 0.705 | 0.797 | 0.875 | 0.689 | 0.771 |
| **Best unsupervised** | 0.897 | 0.625 | 0.737 | 0.730 | 0.510 | 0.600 | 0.827 | 0.650 | 0.728 | 0.702 | 0.609 | 0.652 |
| Our method and its variants | | | | | | | | | | | | |
| **Ours**$_{SubInput}$ | 0.693 | 0.747 | 0.719 | 0.617 | 0.404 | 0.488 | 0.677 | 0.722 | 0.699 | 0.618 | 0.689 | 0.652 |
| **Ours**$_{Concat}$ | 0.877 | 0.707 | 0.783 | 0.734 | 0.637 | 0.682 | 0.895 | 0.699 | 0.785 | 0.752 | 0.645 | 0.694 |
| **Ours** | 0.914 | 0.755 | **0.827** | 0.892 | 0.697 | **0.783** | 0.920 | 0.799 | **0.855** | 0.881 | 0.692 | **0.775** |

## Performance on open domain datasets

Table: Performance comparison of different methods on open domain datasets (including BLESS and Conceptual Graph). We report Precision at rank 200 (P@200), Recall at rank 200 (R@200), F-score at rank 200 (F@200) for Random Splits, and Mean Average Precision (P), Mean Average Recall (R), Mean Average F-score (F) for Lexical Splits. The best performance in the F-score column is boldfaced (the higher, the better).

| Datasets | BLESS | | | | | | Conceptual Graph | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Random Splits (@200) | | | Lexical Splits | | | Random Splits (@200) | | | Lexical Splits | | |
| Method | P | R | F | P | R | F | P | R | F | P | R | F |
| Previous methods | | | | | | | | | | | | |
| **Word2Vec + SVM** | 0.719 | 0.693 | 0.706 | 0.702 | 0.648 | 0.674 | 0.750 | 0.629 | 0.684 | 0.699 | 0.608 | 0.650 |
| **DDM + SVM** | 0.839 | 0.744 | 0.789 | 0.785 | 0.684 | 0.731 | 0.889 | 0.669 | 0.763 | 0.764 | 0.675 | 0.717 |
| **DWNN + SVM** | 0.914 | 0.677 | 0.778 | 0.792 | 0.545 | 0.646 | 0.930 | 0.697 | 0.797 | 0.885 | 0.640 | 0.743 |
| **Best unsupervised** | 0.654 | 0.590 | 0.620 | 0.675 | 0.541 | 0.601 | 0.731 | 0.557 | 0.632 | 0.702 | 0.607 | 0.651 |
| Our method and its variants | | | | | | | | | | | | |
| **Ours**$_{SubInput}$ | 0.675 | 0.659 | 0.670 | 0.621 | 0.600 | 0.610 | 0.683 | 0.623 | 0.652 | 0.608 | 0.572 | 0.589 |
| **Ours**$_{Concat}$ | 0.864 | 0.719 | 0.785 | 0.803 | 0.677 | 0.735 | 0.874 | 0.760 | 0.813 | 0.760 | 0.679 | 0.717 |
| **Ours** | 0.871 | 0.723 | **0.790** | 0.811 | 0.694 | **0.748** | 0.899 | 0.775 | **0.832** | 0.854 | 0.728 | **0.786** |

# Evaluation and Results Analysis

- For domain-specific datasets (i.e., WebIsA-Animal and WebIsA-Plant), on a random split, ours method achieves significantly improvements on the average of F-score by 8.1% and 14.8% compared to the **Word2Vec** and **DDM** methods.
- For open domain datasets (i.e., BLESS and Conceptual Graph), on a random split, our method improves the average F-score by 11.6% compared to **Word2Vec**, by 3.5% compared to **DDM**, and by 2.3% compared to **DWNN** methods.

# Outline

# Evaluation and Results Analysis

**Error Analysis**

- **Inaccurate definition**

  (i) *Terms with biased sense*.

  - Exploring more advanced entity liking techniques, or extract more accurately one from all highly related definitions by combining current context, along with the efficient ranking algorithm.

  (ii) *Prominent context words*. - depending on human-crafted knowledge.

  (iii) *Rare term and entities pairs*, only encoding their term meanings as the definitions in our model.

- **Other relations**

  (i) *Confusing meronymy and taxonomic relations*, e.g., the term pair ("paws", "cat")

  - Adding more negative instances of this kind to the datasets.

  (ii) *Reversed error* (negative instances in WebIsA dataset)

  - Integrating the learning of term embeddings with the distance measure as the feature (e.g., 1-norm distance) into the model.

# Outline

# Conclusions and Future Work

## Conclusions

- We presented a neural network model, which can enhance the representations of term pairs by incorporating their separative accurately textual definitions, for identifying the taxonomic relation of pairs.
- In our experiments, we showed that our model outperforms several competitive baseline methods and achieves more than **82% F-score** on two domain-specific datasets. Moreover, our model, once trained, performs competitively in various open domain datasets. This demonstrates the good generalization capacity of our model.

# Conclusions and Future Work

## Future Work

- One is to consider how to integrate **multiple types of knowledge** (e.g., word meanings, definitions, knowledge graph paths, and images) to enhance the representations of term pairs and further improve the performance of this work.
- The other is to investigate whether this model would be used to the task of multiple semantic relations classification.

## Codes and Datasets

- We will release the codes and datasets at: https://github.com/shengyp/Taxonomic-relation/

Thanks for your time! Any question?