Multiple-source Entity Linking with Incomplete Sources

1st Qinguo Liu
Beijing University of Posts and Telecommunications
Beijing, China
798011435@qq.com

2nd Shui Liu
Tencent Videos
Beijing, China
16237059@qq.com

3rd Lemao Liu
Tencent AI Lab
Beijing, China
lemaoliu@gmail.com

4th Bo Xiao*
Beijing University of Posts and Telecommunications
Beijing, China
xiaobo@bupt.edu.cn

***Abstract*—This paper introduces a new entity linking task from a well-known online video application in industry, where both entities and mentions are represented by multiple sources but some of them may be missing. To address the issue of incomplete sources, it proposes a novel neural approach to model the linking relationship between a pair of an entity and a mention. To verify the proposed approach to this task, it further creates a large scale dataset including 70k examples. Experiments on this dataset empirically demonstrate that the proposed approach is effective over a baseline and particularly it is robust to the missing sources in some extent.**

***Keywords*-deep learning; entity link; entity disambiguation; incomplete Sources;**

I. Introduction

Entity linking aims at grounding a mention describing an entity to the target entity in a given knowledge base, and it plays an important role in many downstream NLP tasks such as language understanding and question answering [1], [2]. In conventional entity linking tasks, where mentions and entities are usually represented by a single source such as a sentence [3], one challenge is that a single source is insufficient to capture different aspects of the meaning for an ambiguous entity [4], [5]. Consequently, it is promising to acquire multiple sources to describe both mentions and entities at various levels of granularity and then build a multiple-source linking model. Unfortunately, in real world scenarios, it is costly to collect a large number of mentions and entities with multiple sources, and it is more practical that some sources for mentions and entities are missing while for others are not [5].

This paper introduces a fresh multiple-source entity linking task from an online video application in industry, where sources for entities and mentions are *incomplete*. A natural solution to such incompleteness is the expectation maximization algorithm which treats a missing source as a latent variable and marginalize it through expectation [6]. However, the missing source in our task is a structure variable (i.e. sentence or picture), which takes value from a large space, and thus it is inefficient to sample a value for approximating its expectation [7].

In this paper, we propose an efficient approach to address the aforementioned issue of data incompleteness. The key idea to our approach is a novel neural model representing the sources in a way such that it is less sensitive to the missing sources. To this end, given a mention and an entity, we firstly represent each pair of sources, one in the mention and the other in the entity, into several vectors and then we employ max-pooling [8] to summarize all those vectors for prediction. Crucially, to make the max-pooling effective, we design a regularization term over the vectors such that their components are comparable. To further verify the proposed approach, we create a large scale dataset [1]including around 70k mention-entity pairs from the log of the video application, each of which is coupled with a label indicating whether the mention is linked to the corresponding entity. We make two-fold contributions as follows:

- We introduce a novel multiple-source entity linking task with incomplete sources from a real world application and create a dataset to facilitate its further research.
- We propose a simple yet effective model to alleviate the issue of data incompleteness, which is robust to the missing sources and delivers substantial improvements over a strong baseline.

II. Task and Dataset

A. Task

Suppose $\mathbf{m} = \langle \mathrm{m}_1, \mathrm{m}_2 \rangle$ denotes a mention represented by two sources m_1 and m_2, $\mathbf{e} = \langle \mathrm{e}_1, \mathrm{e}_2 \rangle$ denotes a 2-source entity, [2] and $\mathcal{KB}$ is a knowledge base $\mathcal{KB}$ consisting of a large number of entities.

Since it is generally cost to collect all sources for mentions and particularly entities in $\mathcal{KB}$ in practice, we propose a new task of **multiple-source entity linking**: linking $\mathbf{m}$ to a particular entity $\mathbf{e}$ in $\mathcal{KB}$, *where some sources are missing for* $\mathbf{m}$ *and entities from* $\mathcal{KB}$. To solve the task, one typically resorts [9], [10], [11] to a binary model $P(z = 1|\mathbf{m}, \mathbf{e}; \theta)$ parametrized by θ, which indicates the probability of linking $\mathbf{m}$ to an entity, and find the target entity according to the following steps:

- Retrieve a subset $\mathcal{E}$ from $\mathcal{KB}$ for $\mathbf{m}$ and then rank the top entity $\hat{\mathbf{e}}$ over $\mathcal{E}$:

$$\hat{\mathbf{e}} = \arg\max_{\mathbf{e} \in \mathcal{E}} P(z = 1|\mathbf{m}, \mathbf{e}; \theta) \tag{1}$$

[1]The dataset will be available at https://github.com once this paper is accepted.

[2]In our dataset there are two sources in total, but our approach in §3 is general to arbitrary number of sources.

978-1-7281-5014-7/19/$31.00

- Link $\mathbf{m}$ to $\hat{\mathbf{e}}$ if $P(z = 1 \mid \hat{\mathbf{e}}, \mathbf{m}; \theta) > 0.5$; otherwise, do not link $\mathbf{m}$ to any entity in $\mathcal{KB}$.

In this way, the entity linking task resembles a binary classification task whose key solution is to build a binary model $P(z|\mathbf{m}, \mathbf{e}; \theta)$ (see §3), and then optimize its parameter θ via maxmimal likelihood estimation over a training dataset as follows:

$$\max_{\theta} \sum_{i} \log P(z^i \mid \mathbf{m}^i, \mathbf{e}^i; \theta) \tag{2}$$

B. Dataset

To establish the new task, we have to provide a dataset, which is used to train our model $P(z|\mathbf{m}, \mathbf{e}; \theta)$ and testify the learned model.

Our dataset consists of a large number of tuples $\langle \mathbf{m}, \mathbf{e}, z \rangle$, which are collected from the log of a large scaled video search website (top 3 video website) in the world. In our dataset, m_1 is the title of the mention and m_2 is a picture, both of which are two sources describing the mention; e_1 and e_2 are a sentence and picture which describe the entity.

In our dataset, each mention $\mathbf{m}$ is provided by a user to the video search website while an entity $\mathbf{e}$ is the search result from a knowledge base maintained by experts for the video application. The label z for the pair $\langle \mathbf{m}, \mathbf{e} \rangle$ is automatically generated through a heuristic method by analyzing the search log for the mention. Our data is collected from the log of a large scaled video search website (top 3 video website) in the world, in which most of the queries can exactly match a certain name of professionally-generated video content (PGC, such as film, cartoon, variety show or tv program). However, certain part of the clicks are on the PGC related user-generated video content (UGC, such as behind the scenes, movie clips, fragment of PGC). Thus, the queries which exactly match the name of PGC, in the task of entity linking, can be considered as name entity, and the clicked UGC can be considered as documents to be linked. To construct a qualified data set, several heuristic rules are applied to mining the query and click video pair from log data, such as the co-occurrence number of query and UGC, the matching rate between query and the name of PGC, the displayed position of UGC, the click though rate (CTR) over PGC, the total number of query. A real sample from the dataset is shown in Table I

Our dataset contains 70k examples, in which positive examples versus negative ones is 2 to 3. The missing rates of m_2, e_1 and e_2 are 0.13%, 51.62%and 0.23%, respectively [3]; and the average lengths of m_1 and e_1 are 22.4 and 251.0. We randomly sample 500 examples from the dataset and manually validate their labels. We find only 3.4% of their labels is not correct, which indicates the quality of our dataset is relatively high. It is worth mentioning that every example in our dataset definitely include the mention title, i.e. m_1, although other sources for $\mathbf{m}$ and $\mathbf{e}$ may be missing.

[3]Because m_2 and e_2 can be automatically extracted from video by programs,but e_1 can only be manually annotated

Table I
A REAL SAMPLE FROM THE DATASET.

label(z)	1
mention title(m_1)	怦然星动票房过亿,曝偶像自白特辑!李易峰戏里戏外男神升级。(Fall in love like a star box office over 100 million,idol confession was exposed! Yifeng Li is a male god both inside and outside the drama.)
mention cover(m_2)	
entity description (e_1)	电影《怦然星动》讲述的是明星与经纪人之间的爱情故事。杨幂饰演的田心与李易峰饰演的苏星宇相识于微时，五年后两人已获事业.....(The movie "fall in love like a star" is about a love story between a star and his agent. Mi Yang as tian xin and Yifeng Li as su xingyu met at a young age, five years after two people have been career.....)
entity cover(e_2)	

III. METHODOLOGY

Our network can be roughly divided into two parts. The first part is the Input Representation. We first complete the embedding of the input text and image, and then extract similarity feature vectors between mention and entity in different dimensions. The second part is Label Prediction. We use a fully connected network with Shared weights to achieve semantic alignment of similarity feature vectors. Then, a robust feature fusion method is used to fuse the four similarity feature vectors. Finally, the fused vectors are fed into the full-connected network to output the results.The overall structure of our network is shown in the Figure 1.

A. Input Representation

On our task, the inputs of our model are mention $\mathbf{m}$ and entity $\mathbf{e}$, both of which include a sentence and a picture as two sources according to our dataset in §2. For simplicity, we suppose the first source (m_1 and e_1) is a sentence while the second one (m_2 and e_2) is a picture. Since the

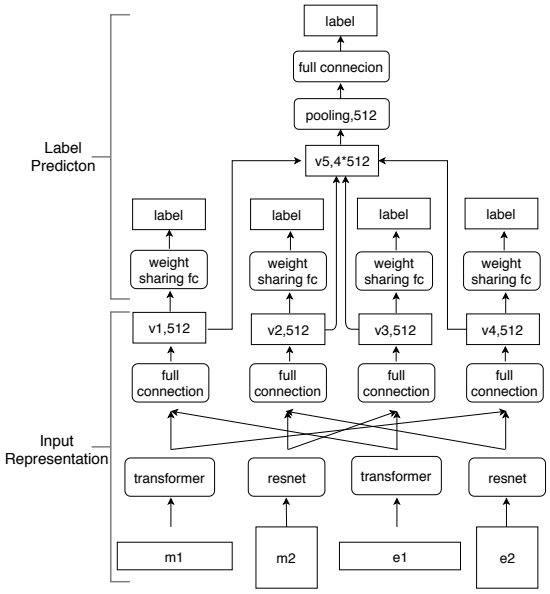


Figure 1. Network structure

sources in $\mathbf{m}$ and $\mathbf{e}$ may be incomplete in practice, we make the following modifications: if the missing source is a sentence, we consider it as a specialized sentence including a single word "$\langle\mathrm{NULL}\rangle$"; if the missing source is a picture, we consider it as a zero matrix with the size identical to that of an ordinal picture. We represent the sentence and picture by using two different networks, i.e. Transformer with multiple-head self attention [12] and Resnet [13].

Specifically, to represent a sentence, we firstly insert a special word $\langle\mathrm{cls}\rangle$ into the beginning of this sentence, then employ a Transformer over the resulting sentence to obtain a matrix, and finally we take the row vector from the matrix with respect to $\langle\mathrm{cls}\rangle$, following BERT [14]. Suppose the representation vectors for sentences m_1 and e_1 are respectively denoted by $\mathrm{T}(\mathrm{m}_1;\theta_{\mathrm{T}})$ and $\mathrm{T}(\mathrm{e}_1;\theta_{\mathrm{T}})$ parametrized by θ_{T}. To represent the picture, we firstly perform a series of operations such as scaling, clipping, standardization, and then we use a Resnet34 model to obtain a vector. Suppose the representation vectors for pictures m_2 and e_2 are respectively denoted by $\mathrm{R}(\mathrm{m}_2;\theta_{\mathrm{R}})$ and $\mathrm{R}(\mathrm{e}_2;\theta_{\mathrm{R}})$ parametrized by θ_{R}. For easier notations, we remove all parameters and denote those four vectors by $\mathrm{T}(\mathrm{m}_1)$, $\mathrm{T}(\mathrm{e}_1)$, $\mathrm{R}(\mathrm{m}_2)$ and $\mathrm{R}(\mathrm{e}_1)$.

Additionally, for capturing the similarity between m and e, we use feedforward neural networks over all pairs of sources from m and e and obtain four new vectors. These new vectors are the final representation vectors of m and e and are denoted as follows:

$$v_1 = \mathrm{F}\big(\mathrm{T}(\mathrm{m}_1), \mathrm{T}(\mathrm{e}_1); \theta_{\mathrm{S}}^1\big), \tag{3}$$

$$v_2 = \mathrm{F}\big(\mathrm{R}(\mathrm{m}_2), \mathrm{R}(\mathrm{e}_2); \theta_{\mathrm{S}}^2\big), \tag{4}$$

$$v_3 = \mathrm{F}\big(\mathrm{R}(\mathrm{m}_2), \mathrm{T}(\mathrm{e}_1); \theta_{\mathrm{S}}^3\big), \tag{5}$$

$$v_4 = \mathrm{F}\big(\mathrm{T}(\mathrm{m}_1), \mathrm{R}(\mathrm{e}_2); \theta_{\mathrm{S}}^4\big), \tag{6}$$

where F is a feedforward network and θ_{S}^i is a parameter for v_i such that all v_i has the same dimension ($i = 1, 2, 3, 4$).

B. Label Prediction

To predict the label z for $\mathbf{m}$ and $\mathbf{e}$, one can summarize all the source representation feature vectors v_1,v_2,$v_3$andv_4 together and then apply a sigmoid function σ over the summarized vector as follows:

$$P(z \mid \mathbf{m}, \mathbf{e}; \theta) = \sigma\big(\theta_w^\top g(v_1, v_2, v_3, v_4)\big) \tag{7}$$

where g is a network to summarize vectors v_i, θ_w is a parameter to weight the summarized vector into a scalar.

To specify the definition g, one natural method is to set g as a fully connected neural network F, which is used as the **baseline** in this paper. Unfortunately, baseline is not robust enough to handle incomplete sources in our task because F is very sensitive to its input v_i for all i.

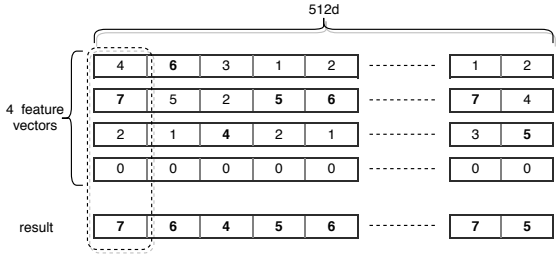


Figure 2. max-pooling instructions

To alleviate this issue, we use max-pooling to summarize v_i, i.e. $g(v_1, v_2, v_3, v_4) = \max(v_1, v_2, v_3, v_4)$, where $\max$ is an element-wise operator over each component of four vectors. Since $\max$ is insensitive to all components in v_i except the largest one, it has the potential to be robust to the missing sources. However, this simple method leads to decreased performance compared with baseline. One possible reason is that each component in different v_i are incomparable by using $\max$. As a result, we build four auxiliary models with shared parameter θ_a which predicts the label using each v_i as follows:

$$P_a(z \mid v_i) = \sigma(\theta_a^\top v_i) \tag{8}$$

Then we train our model regularized by the auxiliary models as follows:

$$\max_\theta \sum_j \log P(z^j \mid \mathbf{m}^j, \mathbf{e}^j; \theta) + \sum_{i=1}^{4} \log P_a(z^j \mid v_i^j) \tag{9}$$

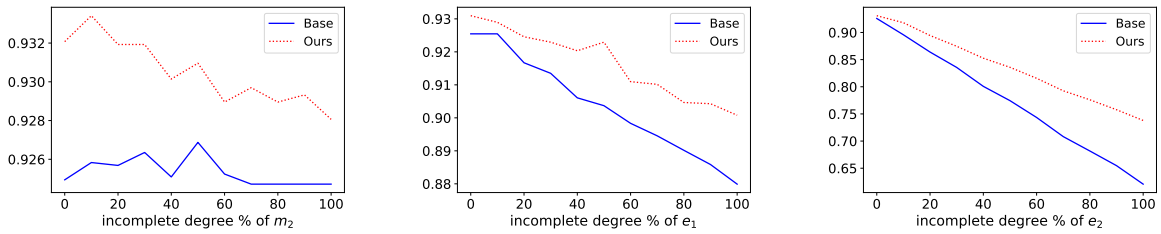
where $P(z \mid \mathbf{m}, \mathbf{e})$ is defined in Eq.(7) with $g = \max$, θ is the overall parameters, and v_i^j is similarly defined in Eq.(6) for the j_{th} example $\langle \mathbf{m}^j, \mathbf{e}^j, z^j \rangle$ in the training set. [4] It is worth mentioning that for testing, we only use the $P(z^j \mid \mathbf{m}^j, \mathbf{e}^j)$ to predict the label.

IV. Experiment

A. Settings

We randomly divide the created dataset into 50k training set, 10k validation set and 10k test set. A two-layer full-connected network including activation function [15]

[4] To train baseline, we set $g = \mathrm{F}$ and then remove the regularization terms in Eq. (9).



(a) (b) (c)

Figure 3. Model accuracy varies with the imperfection of data

and BN [16] is applied to generate the final result. We implement baseline and our models using the Pytorch1.0 framework [17]. All hyperparameters for both models are summarized as shown below. For the sentences(m_1, e_1), they are sampled such that the length is $\leq$ 128 tokens. We use Token Embeddings and Position Embeddings to encode the sequence. The transformer we use is a six-layer encoder network. Each layer of encoder contains a multi-head self-attention and a feed forward network. The representation($\mathrm{T(m_1)}, \mathrm{T(e_1)}$) obtained from transformer is 768-dimensional. For pictures(m_2, e_2), we scale and crop them to 227*227, then use Resnet34 to generate 2048d representation vectors($\mathrm{R(m_2)}, \mathrm{R(e_2)}$). The similarity feature vectors(v_1, v_2, v_3, v_4) we get from feedforward neural networks are all 512d. The θ_w in eq6 is a two-layer full-connected network. The first layer is 512*128, and the second layer is 128*2. There are Relu activation function and BN between the two layers. We initialize the parameters in Transformer and Resent with pretrained models from BERT [14] and Resnet34 [13], and then fine-tune all parameters on our training set. During the training, we set the batch size to 48. We choose SGD [18] with momentum of 0.9 as our optimizer. The learning rates for embedding and other parameters are $1e-4$ and $1e-3$, respectively. The stopping epoch is selected according to the performance on the validation set for both models.

B. Main Results and Analysis

Table II
THE ACCURACY OF MODELS

Methods	Accuracy (%)
Base	91.83
Ours	92.81
w/o Reg	91.61
w/o MP	92.45

The accuracy of our method (Ours) and baseline (Base) on the test set. "w/o Reg" denotes our method without auxiliary regularization and "w/o MP" is our method by replacing max-pooling with a fully connected feedforward network.

Table II summarizes the main results between our models and baseline. It is clear to see that our model gains about 1 points over baseline. In addition, by removing the regularization terms in Eq. (8), the performance is even worse than that of baseline, which shows that making component comparable is critical for our model as mentioned in §3.2. Furthermore, on the basis of fully connected network, regularization gains about 0.4 points over baseline.

Table III
THE ANALYSIS OF SENTENCE LENGTH

Bins	Length	[1,20)	[20,30)	[30, $+\infty$)
	Percents	27.31%	70.36%	2.33%
Methods	Base	90.11	92.45	93.13
	Ours	91.03	93.55	90.56

The accuracy of our method (Ours) and baseline (Base) on each bin of the test set according to the sentence length and number of missed sources in a mention.

As we can see in table III, our model shows better performance on short and medium length sentences, while baseline performs better on long sentences. We speculate that long sentences contain more information, while the full-connected network will encourage that by its additive scoring function. Luckily, such an exception case accounts only for 2.3% and thus it does not decrease the positive effects of our model on average.

C. Robustness

We investigate the robustness of our model by examining the accuracy drops if we artificially remove sources from mentions and entities. To this end, we firstly extract 13.5k examples from the overall data as the new test set, each of which include all four sources; and split the remaining data into training and validation sets, which are used to retrain baseline and our model. Note that for both training and validation sets, their sources may be missing. Then we randomly remove one source for each example in the new test set according to a probability taken from $10\%, 20\%, 30\%, 40\%, 50\%, 60\%, 70\%$ and 100%, leading to 21 test sets with at most one missing source for each example.

Figure 3 depicts the accuracy of both models on the test sets according to randomly remove one source from $\mathrm{m_2}$, $\mathrm{e_1}$ and $\mathrm{e_2}$. [5]We find that our model is more robust to the missing source than baseline as more examples include a missing source. Furthermore, removing $\mathrm{e_2}$ decreases more accuracy compared with removing other sources. This finding suggest that $\mathrm{e_2}$ is more important for making accurate prediction.

[5]We do not conduct experiment on randomly removing $\mathrm{m_1}$, since it always exists in real data.

V. Conclusion

This paper introduced a new entity linking task from industry, where mentions and entities include multiple sources and some sources may be missing. It also presented an efficient and effective approach which is robust to the missing sources for this task. To facilitate research on this task, it created a large dataset. Empirical experiments on this dataset demonstrated the proposed approach delivers gains over a strong baseline.

References

[1] R. Das, M. Zaheer, S. Reddy, and A. McCallum, "Question answering on knowledge bases and text using universal schema and memory networks," *arXiv preprint arXiv:1704.08384*, 2017.

[2] J. Welbl, P. Stenetorp, and S. Riedel, "Constructing datasets for multi-hop reading comprehension across documents," *Transactions of the Association of Computational Linguistics*, vol. 6, pp. 287–302, 2018.

[3] S. Chopra, M. Auli, and A. M. Rush, "Abstractive sentence summarization with attentive recurrent neural networks," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2016, pp. 93–98.

[4] I. Yamada, H. Shindo, H. Takeda, and Y. Takefuji, "Joint learning of the embedding of words and entities for named entity disambiguation," *arXiv preprint arXiv:1601.01343*, 2016.

[5] Y. Eshel, N. Cohen, K. Radinsky, S. Markovitch, I. Yamada, and O. Levy, "Named entity disambiguation for noisy text," *arXiv preprint arXiv:1706.09147*, 2017.

[6] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 39, no. 1, pp. 1–22, 1977.

[7] C. Kong, M. Gao, C. Xu, W. Qian, and A. Zhou, "Entity matching across multiple heterogeneous data sources," in *International Conference on Database Systems for Advanced Applications*. Springer, 2016, pp. 133–146.

[8] Y.-L. Boureau, J. Ponce, and Y. LeCun, "A theoretical analysis of feature pooling in visual recognition," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 111–118.

[9] S. Upadhyay, N. Gupta, and D. Roth, "Joint multilingual supervision for cross-lingual entity linking," *arXiv preprint arXiv:1809.07657*, 2018.

[10] S. Murty, P. Verga, L. Vilnis, I. Radovanovic, and A. McCallum, "Hierarchical losses and new resources for fine-grained entity typing and linking," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018, pp. 97–109.

[11] A. Sil, G. Kundu, R. Florian, and W. Hamza, "Neural cross-lingual entity linking," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[12] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017, pp. 5998–6008.

[13] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[14] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[15] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, 2011, pp. 315–323.

[16] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[17] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.

[18] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.