The proposal and evaluation of Type Estimation for Ontology using Neural Network and Word Embedding

Ryuichiro Hataya Takeshi Morita Takahira Yamaguchi

Keio University

Recently Wikipedia has been a well-used resource for automatic ontologies' constructions. These previous methods have utilized Wikipedia's semi-structured information and the encyclopedia's rich articles have been ignored. In this paper, we propose a type estimation method using distributed representation from all Japanese Wikipedia articles as features, neural network as estimator and class-instance pairs generated by JWO as training data. Our manual evaluation shows more than 60% accuracy for the top one candidate and 80% accuracy for the top three candidates.

1. Introduction

While general ontologies have been regarded as a primary element to create artificial intelligence applications, their construction themselves have been difficult: making them manually like WordNet takes long time and costs a lot, while creating them automatically using natural language processing (NLP) has failed because semantically structured information required for ontologies is too implicit in natural language corpora to extract with NLP.

These days, using Wikipedia's semi-structured information is regarded as practical for automatic extraction of necessary concepts and relations. In fact, these data, for example categories and infoboxes, are used to construct ontologies, for instance YAGO[7], DBPedia[1] and Japanese Wikipedia Ontology(JWO)[8]. However, these previous methodologies have employed only semi-structured parts of pages of Wikipedia but the rest, rich articles are ignored. Plus they cannot deal with exceptional expressions which are found in the anyone-editable encyclopedia.

Word2vec[6] is an empirical methodology to obtain semantically distributed word representation from large text corpora the representation is used for sentiment analysis and text classification as the features. Using this technique, a pair of semantically close words maps to a couple of word vectors close to each other in vector space, that is, their cosine similarity is high, that is close to 1. In addition to this feature, a relation between words becomes a geometrical relation in vector space: $v_{\text{queen}} = v_{\text{king}} - v_{\text{man}} + v_{\text{woman}}$, where $v_\star$ denotes a word $\star$'s word vector.

In this paper, we propose a new methodology for type estimation using Word2vec vectors learned from all Japanese Wikipedia's articles as features and machine generated class-instance pairs from JWO for wide-ranged domain as training data. We regard the task as supervised classification of instances into types instead of prediction of is-a relation vectors. We utilize artificial neural network as classifier in a similar way to [10].

Contact: Hataya, Ryuichiro. hataya@keio.jp

2. Related Works

2.1 Ontology Construction from Wikipedia

Wikipedia is an online encyclopedia which can be edited by anyone. Not just articles, it has semi-structured information such as categories, infoboxes and templates based on articles' types.

Japanese Wikipedia Ontology (JWO) uses Wikipedia's semi-stractured information to construct general ontology. For extraction of is-a relations, it finds candidates of pairs using scraping of list pages and the first sentences of each articles. Then it confirms that the pairs' instances are instances, not classes, with comparison with other semi-stractured information. Wikipedia's articles varies but most first sentences of them share fixed forms as definitions so JWO utilizes these definition sentences for discovering class-instances. For example, the page of "福澤諭吉" starts "福澤諭吉は... 教育者である。". Therefore JWO tags a class "教育者" for an instance "福澤諭吉".

2.2 Related Works Constructing Ontology Using Word vectors

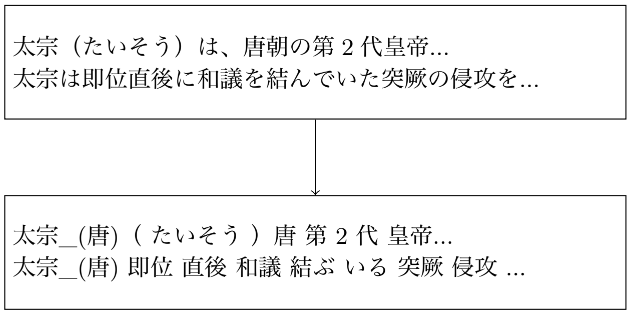
Wohlgenannt and Minic uses analogies of word vectors for extraction of taxonomic pairs with accuracy of 50%. However, [4] shows that hypernym-hyponym relations cannot be found from word vector space generally using analogy or comparison of a pair of word vectors. Therefore, the results in [9] may depend on the domain of words used in the research.

In [2], projection is applied to discriminate whether a relation between a couple of word vectors is a hypernym-hyponym. They made taxonomic and non-taxonomic pairs by hand to learn projections Φ_k where if a pair x, y is hypernym and hyponym in a group k then $|\Phi_k \text{vec}_x - \text{vec}_y|$ is smaller than a threshold.

In [10], Word2vec's vectors learned from Wikipedia's articles are utilized as features to predict their Sekine's extended named entities. Their classifier takes word vectors as inputs and classifies them into the corresponding named entities. The training data are tagged manually.

3. Method

The method is not end-to-end but composed of two parts. In the first part, the Japanese articles are segmented at morpheme level and converted into a distributed representation with Word2vec. The dimensional size of vectors is 200 and all parameters are default values of gensim, a topic model library of Python. Before vectorization, functional words like “は”, “の” are removed and verbs are transformed to dictionary forms for noise reduction. Plus concepts are translated to unique expressions using Wikipedia’s redirect. In Wikipedia’s vocabulary, a concept ideally[1] corresponds to a unique term which is used as unique resource identifier (URI) of the encyclopedia’s articles and Wikipedia’s redirect table resolves these ambiguities. In the case Fig.1, ambiguous 太宗 or 唐朝 is translated to unique 太宗_(唐) or 唐.


太宗（たいそう）は、唐朝の第 2 代皇帝...
太宗は即位直後に和議を結んでいた突厥の侵攻を...

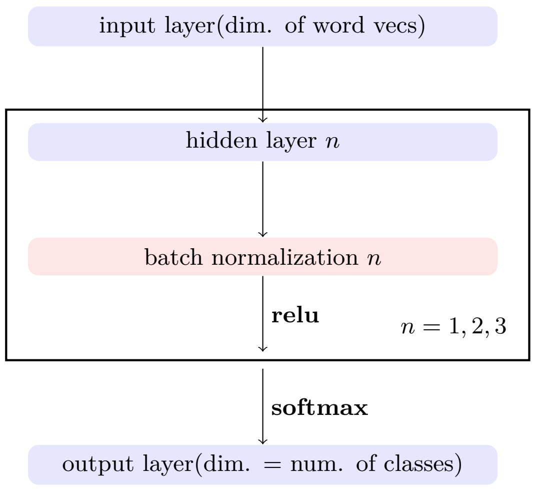
太宗_(唐) (たいそう) 唐 第 2 代 皇帝...
太宗_(唐) 即位 直後 和議 結ぶ いる 突厥 侵攻 ...

Figure 1: segmentation, removal of functional words, transformation of verbs and translation of concepts into unique expression

The next part is a type estimator[2], artificial neural network with three hidden layers (Fig.3.). Each hidden layer is batch-normalized to prevent covariant shifts and every activation function except last softmax function $\frac{\exp(x_i)}{\sum_i \exp(x_i)}$ is ReLU function $\max(0, x)$. The model takes normalized word vectors of instances as input and classify them into classes corresponding to the inputs. For an input instance w_i, the input to the estimator is a word vector v_i and the output vector is a sequence of probabilities $(p_1^i, p_2^i, \ldots, p_n^i)$ and the type is $c_{w_i} = \underset{k}{\operatorname{argmax}}\, p_k^i$. The target for input v_i is a vector y_k^i whose element corresponding to c_{w_i} is 1 and others are 0. The loss function is categorical cross entropy $\frac{1}{N}\sum_i \sum_k y_k^i \ln p_k^i$ and the optimizer is Adam[3].

4. Experiments and Results

We used all Japanese Wikipedia’s articles from October 2016 to learn the feature and 144,000 class-instance pairs generated by JWO 2016 version. The class-instance pairs from JWO include useless ones for example a class which has only one instance as its pair. We deleted such pairs, namely classes whose instances are less than 10 and the



input layer(dim. of word vecs)
hidden layer n
batch normalization n
relu
$n = 1, 2, 3$
softmax
output layer(dim. = num. of classes)

Figure 2: overview of the type estimator

number of class is 1135. The pairs are split into nine to one as training data to testing data. They do not share any instances.

The criterion of evaluation is whether the correct class candidates for the input instance are included in the output candidate(s). We randomly chose 100 outputs for test instances and searched Wikipedia’s articles of each instances to confirm the results.

In this research, top 1 candidate (R1) and top 3 candidates (R3) classes are used.

After 10 epochs learning, the accuracy reaches 60% for R1 and 80% for R3. Tab.1 shows R3 for some input instances.

We randomly sample to downsize the training data from 130,000 to 10,000 then the accuracy decreases from 60% to 58% for R1 and from 80% to 75% for R3 respectively.

4.1 Cross Lingual Type Estimation

A simple application of proposed method is type prediction for other languages’ instances. Previous type estimating methodologies depend on languages’ charactaristcs, for example using of definition in JWO. Hence applying a method for one language to others requires additional sophisticated tuning. However using our method, only a translation matrix W_T or a translation transformation F_T is needed to cross the border between Japanese and another language, where $w_{\mathrm{ja},i} \approx W_T w_{\star,i}$ or $w_{\mathrm{ja},i} \approx F_T(w_{\star,i})$ for i in concept sets C and $\star$ is any given language. For a Japanese instance j, its estimated class is $D(w_{\mathrm{ja},j})$ where D is the proposed estimator. Therefore for a $\star$’s instance $w_{\star,j}$, its class would be $D(W_T w_{\star,j})$ because $W_T w_{\star,j} \approx w_{\mathrm{ja},j}$.

Some examples of type estimation for English instances are shown in Tab.2. We employed Wikipedia’s langlink table to gain Japanese-English concepts’ pairs $\{(i_{\mathrm{ja}}, i_{\mathrm{en}}) | i \in C\}$ and all English Wikipedia articles to learn English word vectors. $W_T = \underset{W}{\operatorname{argmin}} \sum_{j \in C} |W w_{\mathrm{en},j} - w_{\mathrm{ja},j}|^2$ which we minimized with stochastic gradient discent in the same way to [5].

*1 Or may be naïve assumption.
*2 The codes of the estimator implemented with keras, a high-level neural networks library, are available here, https://github.com/moskomule/codes.undergraduate

Table 1: some outputs of the proposed method with their probabilities

instance	1st cand., prob.	2nd cand., prob.	3rd. cand., prob.
学校法人大阪明星学園	学校法人, 0.997	地名, 0.001	学校, 0.000
高ヶ坂	町名, 0.991	地名, 0.008	町, 4.924×10^{-8}
霍英東	政党, 0.179	姓, 0.166	劇作家, 0.130
富小路通	通り, 0.999	町名, 1.587×10^{-7}	坂, 2.998×10^{-8}
ビーちゃん	マスコットキャラクター, 0.860	怪獣, 0.123	キャラクター, 0.007
フランソワ 2 世_(ロレーヌ公)	妃, 0.516	伯爵, 0.174	王族, 0.132

Table 2: some outputs of cross lingual type estimation

instance	1st cand., prob.	2nd cand., prob.	3rd. cand., prob.
Ada_(programming_language)	プログラミング言語,1.0	ソフトウェア, 0.0	コンピュータ, 0.0
Ada_Lovelace	名前,1.0	姓, 0.0029	地名, 0.00011
Yosemite_National_Park	国立公園,1.0	), 0.0	物, 0.0
Barack_Obama	姓,1.0	火砲, 0.0	灯台, 0.0
Kōzan-ji	地名,0.96	寺院, 0.025	都市, 0.0039
Keio_University	学校法人,1.0	地名, 0.0	都市, 0.0

4.2 Comparison with Related Works

Wohlgenannt and Minic uses Word2vec vector as features to find taxonomic relations but their concept domain is small, only terms on global warming. In contrast, our domain is machine generated and large.

In [10], Word2vec and Wikipedia are adopted to label extended named entities for terms. The authors labeled them manually.

5. Conclusion

In this paper, a new type estimation method is proposed, which uses distributed representation as input features and machine-generated class-instance pairs as training data. Manual evaluation shows 60% accuracy for the top one candidate and 80% for the top three candidates.

Further assessment for this is undergoing. This method is expected to estimate hidden types. Some pages of the encyclopedia do not have enough information of types in semi-structured information and instead their types are written in articles implicitly. Word vectors would reflect these hidden information and so the proposed method would enable to estimate classes of these instances.

References

[1] Sören Auer et al. "DBpedia: A nucleus for a Web of open data". In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 4825 LNCS (2007), pp. 722–735. DOI: 10.1007/978-3-540-76298-0_52.

[2] Ruiji Fu et al. "Learning semantic hierarchies: A continuous vector space approach". In: IEEE Transactions on Audio, Speech and Language Processing 23.3 (2015), pp. 461–471. ISSN: 15587916. DOI: 10.1109/TASLP.2014.2377580.

[3] Diederik P. Kingma and Jimmy Lei Ba. "Adam: a Method for Stochastic Optimization". In: International Conference on Learning Representations 2015 (2015), pp. 1–15. arXiv: 1412.6980.

[4] Omer Levy et al. "Do Supervised Distributional Methods Really Learn Lexical Inference Relations?" In: Naacl-2015 (2015), pp. 970–976. URL: http://www.aclweb.org/anthology/N/N15/N15-1098.pdf.

[5] Tomas Mikolov, Quoc V Le, and Ilya Sutskever. "Exploiting Similarities among Languages for Machine Translation". In: arXiv preprint arXiv:1309.4168v1 (2013), pp. 1–10.

[6] Tomas Mikolov et al. "Efficient Estimation of Word Representations in Vector Space". In: (Jan. 2013). arXiv: 1301.3781.

[7] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. "YAGO: A Core of Semantic Knowledge Unifying Wordnet and Wikipedia". In: Proceedings of the 16th international conference on World Wide Web (2007), pp. 697–706.

[8] Susumu Tamagawa et al. "Learning a Large Scale of Ontology fromJapaneseWikipedia". In: 人工知能学会論文誌 25.5 (2010), pp. 623–636. URL: http://ci.nii.ac.jp/naid/130000308072/.

[9] Gerhard Wohlgenannt and Filip Minic. "Using word2vec to Build a Simple Ontology Learning System". In: ISWC2016 Accepted Posters and Demos (2016), pp. 2–5.

[10] 正敏 鈴木 et al. "Wikipedia 記事に対する拡張固有表現ラベルの多重付与". In: 言語処理学会 第 22 回年次大会発表論文集 (2016).