Convolutional Neural Networks for Identify Target Entity Types

ABSTRACT

Detecting entity types that are targeted by the queries help to understand the underlying intent of the queries and also improve the entity retrieval performance. In this paper, we address the task of automatically identifying target types of entity bearing queries. We propose a convolutional neural network approach that can learn important features, without further need to design hand-crafted features required in the learning to rank approaches. Using a standard test collection, we show that our model significantly and substantially outperforms the state-of-the-art probabilistic models and is on par with the learning to rank approach with respect to NDCG@1.

KEYWORDS

Query understanding, query types, entity search, neural network, deep learning

1 INTRODUCTION

A common practice in many search applications is to respond search queries with entities from a knowledge base. Entities, such as people, locations, or organizations, are characterized by their *types*. These types are organized in a hierarchical structure in a knowledge base, referred to as a *type taxonomy*. Identifying the entity types that are the target of search queries is a means towards understanding the underlying intent of the queries [1]. Furthermore, it is shown that target entity types – either being explicitly specified as part of the query, or automatically identified – can improve entity retrieval performance [2, 6, 7, 9].

The primary objective of this paper is to identify target entity types of the queries automatically. Specifically, our goal is to use a type taxonomy to identify "the most specific category of entities that are relevant to the query" [5]. The approaches that are proposed to address the target type identification task can be divided into two categories: (i) probabilistic models, and (ii) learning to rank (LTR) models [5, 6]. Although LTR models provide better performance compared to the probabilistic models, a main shortcoming of these models is that they need several hand-crafted domain-specific features. Our goal is to propose a neural network model that can learn important features, without further need for feature engineering.

Building upon the previous work [5], we develop neural network models for two probabilistic approaches: type-centric (TC) and entity-centric (EC). The type-centric approach identifies entity types using the similarity between the query and term-based representation of a type, while entity-centric approach is based on the top ranked relevant entities to the query. Developing feed-forward neural networks, we show that the performance of these models, while being higher than probabilistic models, is substantially lower than the LTR model. We, therefore, propose convolutional neural network models for both type-centric and entity-centric approaches and further combine them into a single model to capture the most important features of the two regimes.

The main contribution of this paper is to propose a novel convolutional neural network that can automatically identifies target entity of queries. Using a standard test collection, we show that our model significantly and substantially outperforms the probabilistic baselines and is on par with the LTR approach with respect to NDCG@1. This is a remarkable outcome, given the limited amount of available training data and the fact that our model is built based on the minimal hand-engineered features present in the LTR approach; in fact many features, including the taxonomy based features are absent in our approach. The source code and result files of this work will be made publicly available.

2 PROBLEM SPEC AND RELATED WORK

In this section, we first define the problem of Target Identification problem and then we propose the state-of-the-art approaches for solving this problem.

2.1 Problem Definition

Here the objective is to assign target types to queries from a type taxonomy. more formally, we borrow the definition of the task from [5]:

"Find the main target types of a query, from a type taxonomy, such that (i) these correspond to the most specific category of entities that are relevant to the query, and (ii) main types cannot be on the same path in the taxonomy. If no matching type can be found in the taxonomy then the query is assigned a special NIL-type."

2.2 Entity Centric (EC)

The idea of this method is to rank entities based on their relevance to the query, then according to the top-k ranked entities, the target type of query will be determined. This model is similar to late fusion design pattern for object retrieval [11] in which the relevance scores of each type is determined as follows:

$$score_{EC}(t, q) = \sum_{e \in R_K(q)} score(q, e) \times W(e, t) \quad (1)$$

Where R_k is the set of top-k ranked entities for query q and $score(q, e)$ indicates retrieval score of entity e based on language model with Dirichlet smoothing (k=2000). The entity-type association weight, $W(e, t)$ is set uniformly across entities that are typed with t, i.e., $W(e, t) = 1/\sum_{e'} \mathbb{1}(e', t)$, and is 0 otherwise. $\mathbb{1}(e, t)$ is an indicator function that returns 1 if e is typed with t, otherwise returns 0.

WOODSTOCK'97, July 1997, El Paso, Texas USA

ACM ISBN 123-4567-24-567/08/06…$15.00
https://doi.org/10.475/123_4

2.3 Type Centric (TC)

In this method by aggregating descriptions of entities of a specific type, a pseudo document is generated for that type and these documents will be used in ranking. This method is referred to as the early fusion design pattern for object retrieval in [11]. The pseudo frequency of a word for a type is defined as $\tilde{f}(w,t) = \sum_e f(w,e) \times w(e,t)$, where $f(w,e)$ is the frequency of the term w in (the description of) entity e and $w(e,t)$, as before, denotes the entity-type association weight. The relevance score of a type for a given query q is then calculated as the sum of the individual query term scores:

$$score_{TC}(t,q) = \sum_{i=1}^{|q|} score_{LM}(q_i, \tilde{f}) \tag{2}$$

where $score_{LM}(q_i, \tilde{f})$ is the retrieval score of pseudo document of t. We use the same parameter settings as in 2.2.

2.4 Learning to Rank (LTR)

The entity and type centric models capture different aspect of the task and the learning to rank approach proposed in [5] combined these two models in a single framework. In this approach three main feature categories have been used to combine several ranking signals as follows:

(1) *TC and EC features*: this category includes five features which measure the relevancy of query and type based on EC and TC approaches.
(2) *Knowledge base features*: this category includes four features which indicate the properties of the target type in taxonomy (e.g depth, number of children type, etc.).
(3) *Type label features*: this category includes eight features which capture the properties of target type (e.g length of type t in words, etc.)

They employed a random forest algorithm for regression as the supervised ranking method. Their proposed LTR model outperforms EC and TC methods by a remarkable margin but the main problem with this approach is that the features are related to the domain of problem and it can not be easily generalize to other domains such as e-commerce search engines.

3 OUR APPROACH

Our motivation in this paper is to propose a method with minimal feature engineering effort to solve target type identification. For this task, we use the test collection introduced in [5]. In this test collection, for a given query there is a pool of target entity types. for each target type in the pool, there is a label between zero (non-relevant) and seven (the most relevant) which indicates the relevance degree of the query and that target type. Similar to the proposed LTR approach in [5] we cast the type identification problem into regression problem i.e for a given pair of query and type the goal is to predict the relevancy score.

Following the idea of TC and EC method introduced in section 2.3 and 2.2 we defined two matrices i.e I_{TC} and I_{EC} that models word level similarities via word embeddings.

$\boldsymbol{I_{TC}}$: For a given query-type pair (q,t), we first extract top-50 words representing type t. In order to extract these words, we concatenate all abstracts of entities associate with that type and make a pseudo document for each type. We calculate the tf.idf according to these pseudo documents. The words are sorted according to the tf.idf score. We then build I_{TC} matrix as follows. Each row of matrix indicates a query term (e.g q_i) of q (in order of occurrence) and each column of the matrix indicates a representing word (e.g w_j) of that type (sorted by tf.idf score). The entry $I_{TC}[i,j] = cos(\vec{q_i}, \vec{w_j})$ where $\vec{q_i}$ and $\vec{w_j}$ indicates the pre-trained word embedding vectors of [8].

$\boldsymbol{I_{EC}}$: For a given query-type pair (q,t), we first used a language retrieval model to retrieve top-20 ranked entities (i.e $e_1, \cdots, e_{20}$) for query q. we treat the abstract of each entity as a document and use language modeling with Dirichlet smoothing (k=2000) to retrieve these entities.

I_{EC} is made by concatenation of twenty matrices as $I_{EC} = [I_{E_1C}, \cdots, I_{E_{20}C}]$ where I_{E_iC} is a matrix indicating the relevancy of query q and entity e_i considering the type t. Each entry of matrix I_{E_iC} is computed as follows:

$$I_{E_iC}[m,n] = \begin{cases} cos(q_m, w_{i,n}) \times W(e,t) & \text{type(e)} \cap t \neq \phi \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

Where q_m is the m^{th} query term and $w_{i,n}$ $(1 \leq n \leq 20)$ indicates the n^{th} word (sorted on tf.idf score) representing the entity e_i. $W(e,t)$ is the function described in section 2.2 and $type(e)$ is the set of types associated with entity e.

In our first attempt to solve the problem we use two feedforward neural networks. the input of these networks are I_{TC} and I_{EC}, respectively.

Similar to the translation, rotation and scale invariance of images in CNN, CNN is able to learn the local features from words or phrases in different places of text [10]. As a result, in our second attempt, we present a jointed convolutional neural network and feedforward (FF) architecture that takes the local features extracted by CNN as input to FF for target type identification problem. We try this architecture twice with I_{EC} and I_{TC} as the inputs of the CNN models, respectively.

Finally, following the idea of combining entity centric and type centric models[2, 5], we propose the architecture represented in fig.1. This architecture is the combination of the above mentioned CNN networks. In this figure, I_{TC} is the input of type centric part of network with size of (14×50) and I_{EC} is the input matrice of entity centric part with size of (14×400) (14 is the maximum size of the queries). The number of filters and their size is written under each layer. The output node predict target type identification score between zero and seven.

4 MODEL IMPLEMENTATION DETAILS

This section describes the configuration of the FF and CNN Models.

Model Training is performed using 5 fold *nested cross-validation* [3] where model hyperparameters are selected for each test fold based on standard CV experiments on the training folds.

Model Optimization use the Adam optimizer with batch size 128, learning rate 0.0001. We implemented our model using Keras [4].

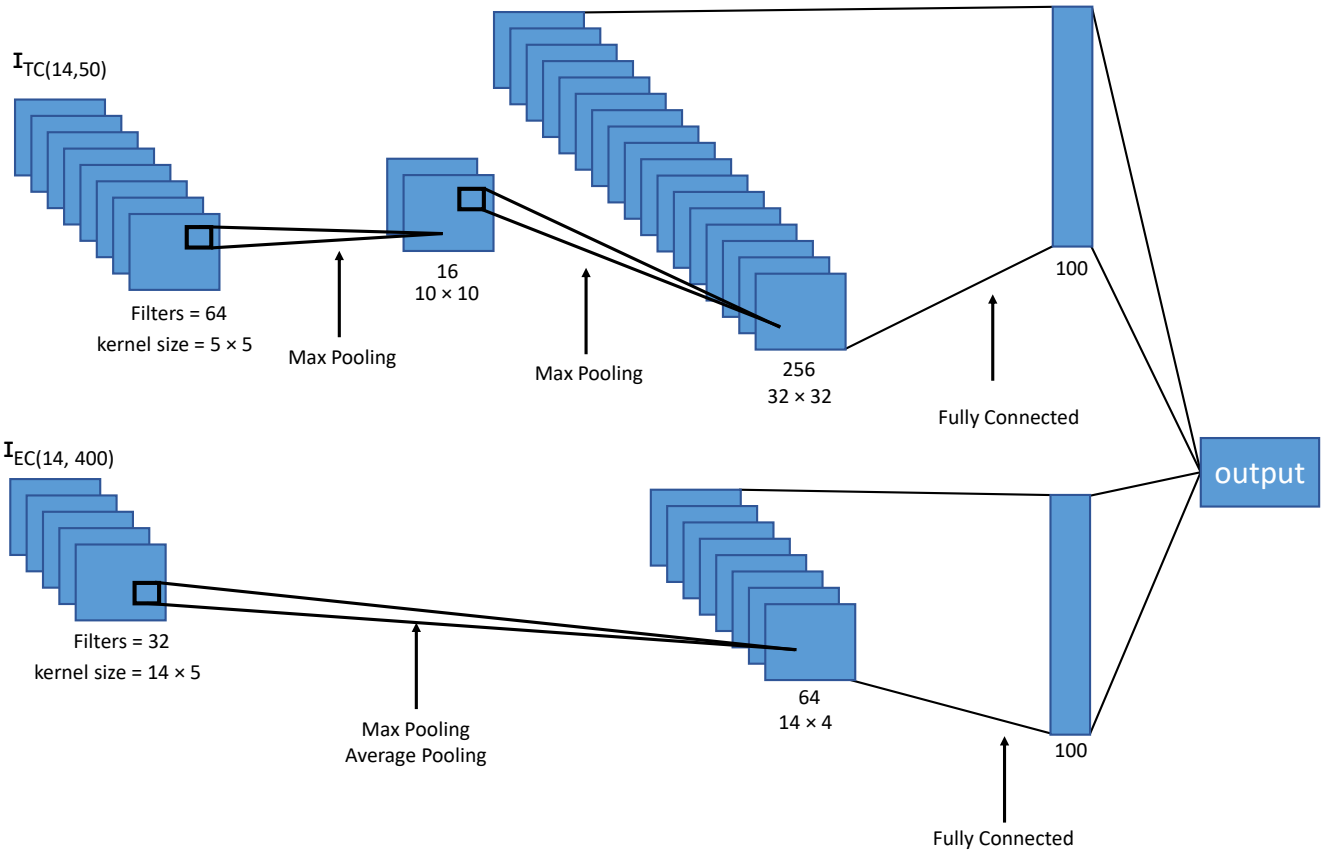

Figure 1: propose model structure

Table 1: Target type detection performance. Significant improvements over probabilistic, feedforward models are marked with *, ** respectively.

Feature Group	Retrieval model	NDCG@1	NDCG@5
Entity Centric	Probabilistic	0.1490	0.3223
Type Centric		0.2341	0.3780
Entity Centric	FeedForward	0.1886*	0.3289
Type Centric		0.2786*	0.4011*
Entity Centric	CNN	0.3341***	0.4586***
Type Centric		0.4034***	0.5287***
Entity + Type	CNN	0.4575***	0.5994***
Entity + Type + Knowledge Base + Type Label	LTR	0.4842***	0.6355***

Model Structure model structure of CNN is depicted in fig.1 The feedforward model of type centric approach is a three-layer network with [1000, 1000, 5000] neurons. The activation function is relu. the dropout parameter is selected to be 0.3. The feedforward model of entity centric approach is a one-layer network with 1000 neurons. The activation function is Relu. the dropout parameter is selected to be 0.

5 RESULT AND DISCUSSION

Table 1 presents the evaluation results. We find that the feedforward network significantly (except Entity Centric for NDCG@5 measure) outperforms the probabilistic model both in the entity and type centric approaches. significant improvement is measured with p<0.05 using two-tailed pair-test. In addition, the CNN architecture improves the feedforward network significantly by a large margin both in entity centric and type centric approaches. Another interesting observation here is that the hybrid model (i.e entity + type centric) can improve the NDCG@1 and NDCG@5 significantly.

The interesting observation here is that the NDCG@1 of CNN model (in hybrid setting) is not worse than the LTR model significantly. In other words, although, the LTR model utilizes several hand-crafted features, it can not outperform the CNN proposed model.

In order to better analyze the behavior of CNN network, we visualized the input of I_{TC} and the output of first convolution layer for the query *"Give me a list of all trumpet players that were bandleaders."* and type *"MusicalArtist"* in fig.2 and fig.3 respectively. As indicated in these figures the horizontal axis indicates top-50 words representing *MusicalArtist* type sorted by tf.idf from left to right. in this figure, lighter entries indicate the larger value of similarity between a query term and the corresponding representing term. By comparing the output and output matrices, we find that the CNN model is able to remove noises i.e non-relevant terms to that specific type. for example the weights associated with query terms "*give*" and "*me*" are reduced significantly. According to this observation, CNN network outperforms the FF because it can extract local

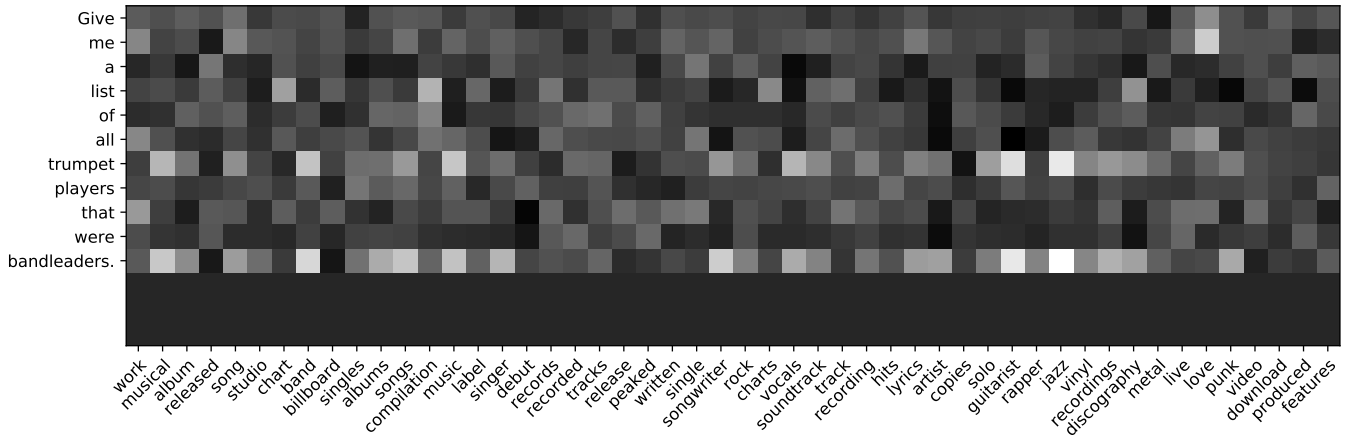

Figure 2: input of type centric CNN model

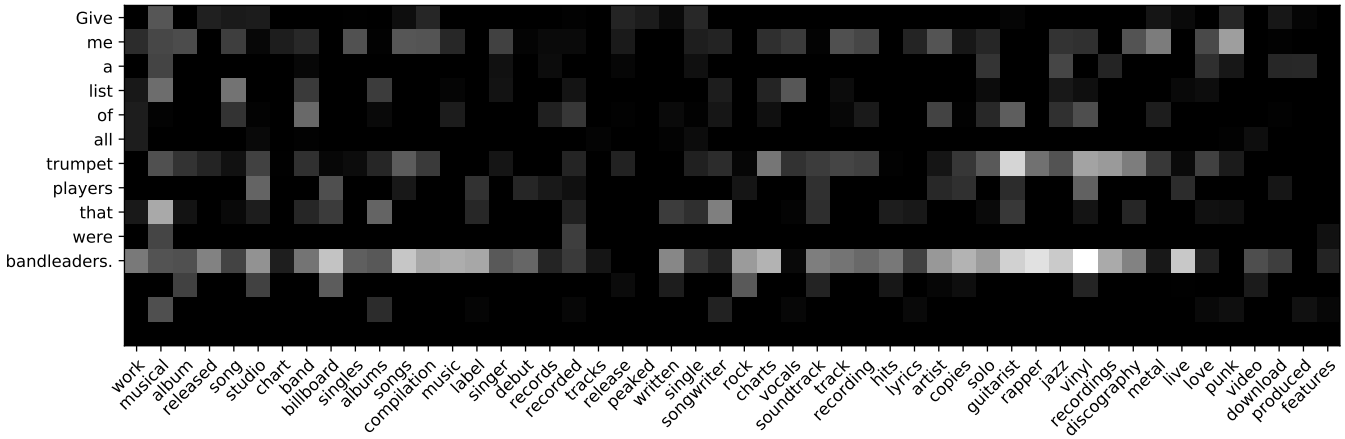

Figure 3: output of type centric CNN model after conv1 layer

features (by local we means the feature associated with the context of query) while the FF network only uses the word2vec embedding of each word and that not consider it's context in the query.

6 CONCLUSION

Target type identification is an important problem in entity retrieval, it can enhance the performance of entity retrieval for entity-bearing queries. In this paper we propose a convolutional neural network architecture to capture local features of entity bearing queries. Our experiments indicate that the proposed architecture outperform simple feedforward network as well as probabilistic (TC, EC) models. Our architecture considers the order of query terms as well as the order of top-k terms representing entities and types. several layers of CNN extract important features to solve this problem. Our interesting observation is that LTR model with a rich set of hand-crafted features can only slightly perform better than the CNN.

REFERENCES

[1] Krisztian Balog. 2018. *Entity-Oriented Search*. The Information Retrieval Series, Vol. 39. Springer. https://eos-book.org

[2] Krisztian Balog, Marc Bron, and Maarten De Rijke. 2011. Query Modeling for Entity Search Based on Terms, Categories, and Examples. *ACM Trans. Inf. Syst.* 29, 4 (Dec. 2011), 22:1–22:31.

[3] Gavin C Cawley and Nicola LC Talbot. 2010. On over-fitting in model selection and subsequent selection bias in performance evaluation. *Journal of Machine Learning Research* 11, Jul (2010), 2079–2107.

[4] François Chollet et al. 2015. Keras. https://keras.io. (2015).

[5] Darío Garigliotti, Faegheh Hasibi, and Krisztian Balog. 2017. Target Type Identification for Entity-Bearing Queries. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '17)*. ACM, New York, NY, USA, 845–848.

[6] Darío Garigliotti, Faegheh Hasibi, and Krisztian Balog. 2018. Identifying and Exploiting Target Entity Type Information for Ad Hoc Entity Retrieval. *Information Retrieval Journal* (05 Dec 2018). https://doi.org/10.1007/s10791-018-9346-x

[7] Rianne Kaptein and Jaap Kamps. 2013. Exploiting the Category Structure of Wikipedia for Entity Ranking. *Artif. Intell.* 194 (Jan. 2013), 111–129.

[8] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and Their Compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*. Curran Associates Inc., 3111–3119.

[9] Jovan Pehcevski, James A Thom, Anne-Marie Vercoustre, and Vladimir Naumovski. 2010. Entity ranking in Wikipedia: utilising categories, links and topic difficulty prediction. *Information Retrieval* 13, 5 (2010), 568–600.

[10] Xingyou Wang, Weijie Jiang, and Zhiyong Luo. 2016. Combination of convolutional and recurrent neural network for sentiment analysis of short texts. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. 2428–2437.

[11] Shuo Zhang and Krisztian Balog. 2017. Design patterns for fusion-based object retrieval. In *European Conference on Information Retrieval*. Springer, 684–690.