Entity Attribute Relation Extraction with Attribute-Aware Embeddings

Dan Iter*
Stanford University
daniter@stanford.edu

Xiao Yu
Google
yux@google.com

Fangtao Li
Google
lifangtao@google.com

Abstract

Entity-attribute relations are a fundamental component for building large-scale knowledge bases, which are widely employed in modern search engines. However, most such knowledge bases are manually curated, covering only a small fraction of all attributes, even for common entities. To improve the precision of model-based entity-attribute extraction, we propose attribute-aware embeddings, which embeds entities and attributes in the same space by the similarity of their attributes. Our model, EANET, learns these embeddings by representing entities as a weighted sum of their attributes and concatenates these embeddings to mention level features. EANET achieves up to 91% classification accuracy, outperforming strong baselines and achieves 83% precision on manually labeled high confidence extractions, outperforming Biperpedia (Gupta et al., 2014), a previous state-of-the-art for large scale entity-attribute extraction.

1 Introduction

Modern search engines often attempt to provide structured search results that reveal more facets of the search query than explicitly requested. These results rely on knowledge bases that contain tuples of the form (*entity*, *attribute*, *value*). However, the number of known entities and attributes in these knowledge bases is limited and there is a long tail of both entities and attributes that is too large to be manually curated. The goal of automatic entity-attribute extraction is to replace manual knowledge acquisition which is expensive and biased towards popular entities (Bollacker et al., 2008; Dong et al., 2014). Previous studies have proposed model-based approaches that use various NLP features, distant supervision and traditional machine learning methods for entity-attribute extraction but their precision has not been high enough to replace manually curated knowledge bases (Auer et al., 2007; Carlson et al., 2010; Gupta et al., 2014).

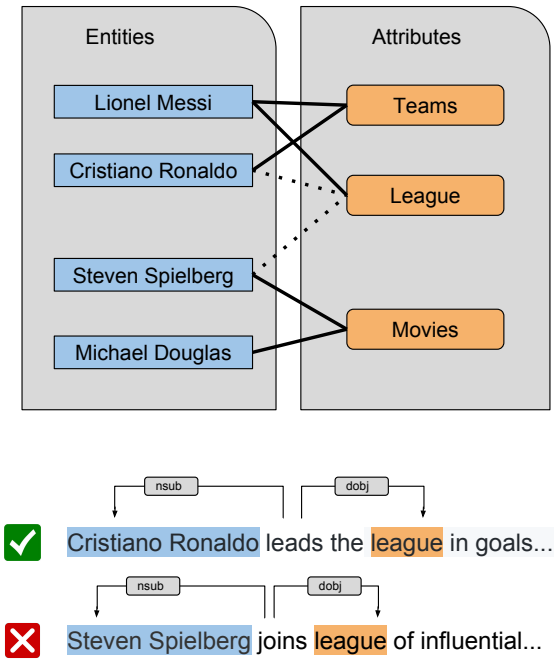

Figure 1: The top figure shows an example of known attributes (solid lines) and candidate attributes (dotted lines). Below are support sentences for the candidate entity-attribute pairs. The attributes of related entities can be used to improve entity representation and entity-attribute classification accuracy.

The key insight of this paper is that entities that share many attributes are often similar. This is an extension of the distributional hypothesis, (Harris, 1954; Weeds and Weir, 2003), which states that words with similar semantic meanings tend to appear in similar contexts, and builds on work that use referential attributes to estimate semantic relatedness (Gupta et al., 2015; Freitas et al., 2013). For the attribute-aware embeddings, we argue that a good representation for an entity can be

* Work done during an internship at Google.

Proceedings of Deep Learning Inside Out (DeeLIO):
The First Workshop on Knowledge Extraction and Integration for Deep Learning Architectures, pages 50–55
Online, November 19, 2020.

inferred from its most common attributes, which we may have access to from an external source of knowledge. In Figure 1, we want to classify two candidate relations given the other known relations. If the model has previously seen a link between *Teams* and *League* but not *Movies* and *League*, we can correctly predict that *Ronaldo* should have the *League* attribute and *Speilberg* should not, despite the surface form in the sentences being similar. We generalize this intuition by learning embeddings for each entity based on the most common attributes for that entity.

We propose EANET, a neural model that combines a path-embedding model from dependency parse trees, with **attribute-aware embeddings** which we describe in this paper. The proposed model captures both the dependency path for a given sentence between potential entity and attribute candidates, as well as a distributional representation for both entities and attributes based on an attribute distributional assumption. Our model learns general representations for specific entities with few mentions by learning embeddings based on attributes for those entities observed in the distant supervision.

2 Methods

2.1 Task Definition

The objective of this work is to determine, for a given term-pair (e, a), if a is an attribute of the entity e. In order to classify entity-attribute term pairs, we have access to a multiset of sentences $\mathcal{S}^{(e,a)}$ where the terms co-occur. These sentences only capture local information about how the entity and attribute terms relate at the sentence level. Our model also has access to global information from the set of known entity-attribute pairs from the training data. For each term-pair, (e, a), we are given a set of known true entity-attribute pairs $\mathcal{K}^e$ and $\mathcal{K}^a$, where $\mathcal{K}^e$ is a set where the entity is always e and $\mathcal{K}^a$ is a set where the attribute is always a. We learn a binary classifier on the input of entity-attribute pairs, their support sentences, and their known neighbors.

2.2 Path Embedding Model

Our baseline model is inspired by a hypernym classification model proposed by Shwartz et al. (2016), also using a pair of terms with a set of support sentences where the terms co-occur.

Sentences are represented with their shortest dependency path between two candidates, as proposed in (Fundel et al., 2006). The bottom of Figure 1 shows an example of the shortest path between an entity and attribute for one sentence. We converts each sentence from a string to a list of terms, where the first and last term is either the entity or the attribute. Each term in the dependency path is represented by the lemma of the term, the part-of-speech tag, the dependency label, the direction of the dependency path to the parent (left, right or root). Each of these features is embedded and concatenated to produce a sequence of vectors that represents the dependency path. The concatenation is the edge representation $\overrightarrow{v}_{edge} = [\overrightarrow{v}_{lemma}, \overrightarrow{v}_{pos}, \overrightarrow{v}_{dep}, \overrightarrow{v}_{dir}]$

The sequence of terms in each path is input into an LSTM to produce a single vector representation for the sentence, $\overrightarrow{v}_s$. This is repeated for each sentence producing one vector per sentence. The sentences are aggregated with a weighted mean of the sentence representations to form a representation of the multiset of sentences, $\overrightarrow{v}_{sents(e,a)}$.

2.3 Distributional Representation

As proposed by Shwartz et al. (2016), adding the word embedding or distributive representation for the candidate strings can improve the performance of the model. The embeddings of the two candidate terms in the entity-attribute pair are concatenated to each side of the aggregated sentences vector described in the previous section. The embeddings for e and a are simply $\overrightarrow{v}_e$ and $\overrightarrow{v}_a$. Thus the full representation of a entity-attribute pair is: $\overrightarrow{v}_{(e,a)} = [\overrightarrow{v}_e, \overrightarrow{v}_{sents(e,a)}, \overrightarrow{v}_a]$

2.4 Attribute-Aware Embeddings

We propose to use an attribute-aware representation in the classification model to leverage "similarity" between terms. In the entity-attribute relation setting, a strong signal of what entities are similar is how their attributes overlap. Therefore, we create an **attribute-aware embedding** for each entity and attribute that captures term similarity by shared attributes, rather than relying on *word embeddings*, which are learned from terms being in similar contexts. This helps to generalize to new unseen data for which we may know some related entities and attributes. Figure 2 illustrates the model and shows how the representations are combined in the classifier.

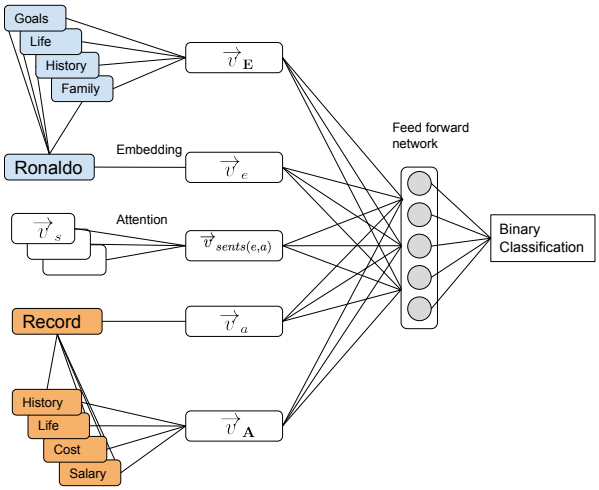

Figure 2: Model for binary classification of a pair of entity attribute candidates. For each entity and attribute, related attributes are aggregated into attribute-aware embeddings that are concatenated to the sentence and distributional representations.

2.5 Learning Attribute-Aware Embeddings

Entities are modeled as a weighted sum of their attributes. Similarly, attributes can be represented by the entities that they describe, which would create a symmetry in the model. However, empirically we found that attributes provide more signal so we use attributes to represent other attributes. Given an attribute, we can find all entities in the training data that have that attribute. We then find all the attributes for each entity. We take the ranked set of attributes by count (of entities) to associate with the attribute.

We initialize attribute embeddings with Glove (Pennington et al., 2014) word embeddings. For an entity, we take all the known attributes from $\mathcal{K}^e$. The representation of each entity is the weighted sum of the known attributes, with learned attention weights. The weights are shared between entities and attributes.

We concatenate these vectors to the full representation for the entity-attribute term-pair: $\overrightarrow{v}_{(e,a)} = [\overrightarrow{v}_{\mathbf{E}}, \overrightarrow{v}_{e}, \overrightarrow{v}_{sents(e,a)}, \overrightarrow{v}_{a}, \overrightarrow{v}_{\mathbf{A}}]$

In Figure 2 the 5 concatenated vectors are shown vertically in the middle and are input into a feed forward neural network, followed by a sigmoid layer and a logistic loss.

3 Dataset Creation

3.1 Dataset

We sample sentences from a subset of online news articles and label them with our distant supervision knowledge base (Mintz et al., 2009) using a query streams as the source of supervision, as in (Paşca et al., 2007). We sample 12.6 million entity-attribute pairs from a knowledge base, finding 6 million unique entities and 788 thousand unique attributes. Each sentence that contains an entity-attribute pair from our knowledge base is used as the support set for that pair. All pairs with less than 30 support sentences are discarded leaving 351 thousand distantly labeled positive examples and 14 million negatively labeled examples. Negative examples are sentences that contain a known entity and a known attribute but the entity is not annotated with this attribute in our knowledge base. We split the positive examples into roughly 75% train, 20% test and 5% validation. Negative examples are randomly sampled so we sample equivalent amounts for test and validation but use about 14 million negative examples during training, weighing the two classes accordingly.

3.2 Annotations and Labels

True entity-attribute relations are sampled from our knowledge base. Negative examples of entity-attribute pairs are combinations of entities and attributes that appear together in sentence but are not in our knowledge base. We augment the negative examples with randomly sampled noun phrases (including modifiers) from sentences in our corpus. We also flip the order of true entity-attribute pairs and use them as negative examples because the relation is not symmetric. Support sentences for each pair are found by exact match of both terms.

Known attributes ($\mathcal{K}^e$) for each entity and known entities ($\mathcal{K}^a$) for each attribute are the top 20 most common attributes for that entity or attribute in the training data by the count of support sentences. If there are no known attributes for an entity, as is the case for our test data in the entity-split setting, we use the top 20 known attributes by co-occurrence with the entity in the support sentences.

4 Experiments

We evaluate our model on two basic metrics. (1) The ability of the model to fit the data and generalize to a held-out test set which is measured

	Entity Split	Random Split
Path	0.784	0.821
Dist.	0.772	0.882
Attr-Aware	**0.825**	0.905
EANET	0.824	**0.913**

Table 1: Accuracy of two baselines and two models with attribute-aware embeddings (in bold) for entity split and random split test datasets.

by accuracy on the test set. (2) The ability of the model to extract high quality entity-attribute pairs which we evaluate with manual evaluation on a small set of extracted pairs.

4.1 Entity Attribute Classification

We compare two variations of our model to two strong baseline models that are also based on word and path embedding neural network models.

The baselines are the **Path** and **Dist.** (ie. distributional) models described in Section 2.2 and Section 2.3 respectively. The **Attr-Aware** model (ie. attribute-aware), uses both the path embeddings and our attribute-aware embeddings but does not include the word embeddings of the terms.

We evaluate two regimes; entity split and random split. Entity split separates test and train by entity so there is no overlap in entities. The random split naively splits the set of all entity-attribute pairs. In the random split, every entity-attribute pair is unique so for every candidate pair in the test set, the model has never seen any sentences where the entity and attribute appeared together during training. For example, from Fig 1, the training data may contain (Ronaldo, Teams) and (Spielberg, Movies) and the test data will have (Ronaldo, League) and (Spielberg, League). In this setting, we have some learned representation of the entity and the attribute separately but have not observed them together in the training data. The sentences used for the mention level representations are also split between training and test sets.

Table 1 shows the results for the two settings and the accuracy for each of the two baselines and the two proposed models described above. Models that use our proposed attribute-aware embeddings outperform the baselines in every setting. EANET achieves the highest accuracy on the random test split at 91.3%.

(Common)	Precision	F1
Biperpedia	0.547	0.707
EANET	0.837	0.911

Table 2: Precision and F1 for EANET and Biperpedia on 1000 extracted entity-attribute pairs each that were manually labeled.

4.2 Entity Attribute Extraction

We present results for two evaluations with human labels; (1) we compare extracted entity-attribute pairs to those extracted by the previous state-of-the-art Biperpedia (Gupta et al., 2014) and (2) we report precision over a small set of longtail entity-attribute pairs that did not appear in our distant supervision.

Biperpedia: First we sample 20 entities from the most common entities in the test data. We sample the entities such that each entity belongs to a different knowledge graph class to fairly compare against Biperpedia which extracts attributes at the class level rather than the entity level. For each entity, we randomly sample 50 attributes extracted by EANET. We use the same entities for Biperpedia and sample 50 attributes each from those extracted by Biperpedia. 1000 high-confidence examples are extracted using each model and they are manually labeled.

Since Biperpedia extracts attributes at the class level (eg. *Country* instead of *USA*), the attributes are sampled from the class of the entity. The comparison is not fair because many errors come from the mapping from entity to class (eg. *Countries* have *Prime Ministers* but the *US* does not). Nevertheless, this is a good evaluation for how relevant attributes from Biperpedia are for a given queried entity.

From a total of 1000 manually labeled entity-attribute pairs from each model, EANET achieves 83.7% precision while Biperpedia achieves only 54.7% precision. Table 2 shows precision and F1 scores for EANET and Biperpedia for the 1000 entity-attribute pairs extracted for each model conditioned on the same 20 entities. The improvement in EANET comes from both more fine-grained typing, using entity level attributes rather than class level, and from higher precision classifications.

(Longtail)	Precision	F1
EANET	0.623	0.768

Table 3: Precision and F1 for EANET on 1000 extracted longtail entity-attribute pairs that were manually labeled. Neither the entities nor the attributes in these pairs appeared at all in our distant supervision.

Longtail: A goal of EANET is extracting longtail entity-attribute pairs that would likely be missed by human created knowledge bases. We manually labeled the top 1000 entity-attribute pairs by frequency where **neither** entity nor attribute appear in the distant supervision. EANET achieves 62.3% precision on 1000 manually labeled entity-attribute pairs. We expected the precision to be worse because the "known" attributes used for the attribute-aware embeddings are much noisier when neither the entity nor the attribute has been seen in any positive training data.

4.3 Error Analysis

For more insight into the performance of our model, we analyze the type of errors that our model makes and discuss possible trade-offs and possible future improvements.

Error Types. From manually annotating a small set of examples, we find that there are 6 general types of errors:

- x of y—a common pattern for entity-attribute pairs such as "the height of a person" but also often occurs with non-entity attribute pairs such as "a lot of people",
- IsA relationships,
- extracting the value rather than the attributes—extracting the name of a drug rather than the term "medication",
- incorrect entity extraction—extracting "medication" as an attribute for "heart" instead of "heart disease",
- general attribute—some terms such as "number" and "direction" are often attributed to rarer entities because they seem generic and similar to other common attributes and
- other miscellaneous errors.

In the comparison to Biperpedia we analyze the type of errors our model makes with common extractions. In this setting, most entities are frequent in the dataset. The most common errors are x of y, value extraction and incorrect entities. In all cases, it seems that generally high co-occurrence between terms is often a contributor to false positives and these are the most common cases of non-entity-attribute co-occurrence in the training data.

High Confidence		
Label	Entity	Attribute
True	Britain	invasions
True	homeless	medical care
False	prostate cancer	Metformin
False	petroleum	migration
Longtail		
Label	Entity	Attribute
True	medical students	rotations
True	endophthalmitis	injections
False	SXSW	festival
False	third party	operating systems

Table 4: A few positive and negative examples from common and longtail extractions.

In the longtail evaluation, both the entities and attributes are relatively rare and do not occur in any of our training data. In this case, we see many IsA relationships and general attributes. The former could likely be remedied by using supervision of IsA relationships to generate negative training data. The latter is an overgeneralization by the model.

Some examples of correct and incorrect extractions are shown in Table 4.

5 Conclusion

We present EANET, a neural model for entity-attribute relation extraction from text. We describe a mechanism for learning attribute-aware embeddings for entities and attributes in our training data that capture similarities between entities by embedding the similarities between their known attributes. We show that our model outperforms the previous approaches for entity-attribute relation extraction and that it can be used to learn representations for longtail entities.

References

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. Dbpedia: A nucleus for a web of open data. In *The semantic web*, pages 722–735. Springer.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collab-

oratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250. AcM.

Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R Hruschka, and Tom M Mitchell. 2010. Toward an architecture for never-ending language learning. In *Twenty-Fourth AAAI Conference on Artificial Intelligence*.

Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. 2014. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 601–610. ACM.

André Freitas, João Gabriel Oliveira, Seán O'riain, João CP Da Silva, and Edward Curry. 2013. Querying linked data graphs using semantic relatedness: A vocabulary independent approach. *Data & Knowledge Engineering*, 88:126–141.

Katrin Fundel, Robert Küffner, and Ralf Zimmer. 2006. Relex—relation extraction using dependency parse trees. *Bioinformatics*, 23(3):365–371.

Abhijeet Gupta, Gemma Boleda, Marco Baroni, and Sebastian Padó. 2015. Distributional vectors encode referential attributes. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 12–21, Lisbon, Portugal. Association for Computational Linguistics.

Rahul Gupta, Alon Halevy, Xuezhi Wang, Steven Euijong Whang, and Fei Wu. 2014. Biperpedia: An ontology for search applications. *Proceedings of the VLDB Endowment*, 7(7):505–516.

Zellig S. Harris. 1954. Distributional structure. *Word*, 10:146–162. Reprinted in J. Fodor and J. Katz, *The Structure of Language*, Prentice Hall, 1964 and in Z. S. Harris, *Papers in Structural and Transformational Linguistics*, Reidel, 1970, 775–794.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 1003–1011. Association for Computational Linguistics.

Marius Paşca, Benjamin Van Durme, and Nikesh Garera. 2007. The role of documents vs. queries in extracting class attributes from text. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 485–494. ACM.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Vered Shwartz, Yoav Goldberg, and Ido Dagan. 2016. Improving hypernymy detection with an integrated path-based and distributional method. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2389–2398, Berlin, Germany. Association for Computational Linguistics.

Julie Weeds and David Weir. 2003. A general framework for distributional similarity. In *Proceedings of the 2003 conference on Empirical methods in natural language processing*.