Record2Vec: Unsupervised Representation Learning for Structured Records

Adelene Y.L. Sim
Amazon.com, Inc.
Seattle, WA, USA
adelenes@amazon.com

Andrew Borthwick
Amazon.com, Inc.
Seattle, WA, USA
andborth@amazon.com

***Abstract*—Structured records – data with a fixed number of descriptive fields (or attributes) – are often represented by one-hot encoded or term frequency-inverse document frequency (TF-IDF) weighted vectors. These vectors are typically sparse and long, and are inefficient in representing structured records. Here, we introduce Record2Vec, a framework for generating dense embeddings of structured records by training associations between attributes within record instances. We build our embedding from a simple premise that structured records have attributes that are associated, and therefore we can train the embedding of an attribute based on other attributes (or context), much like how we train embeddings for words based on their surrounding context. Because this embedding technique is general and does not assume the availability of any labeled data, it is extendable across different domains and fields. We demonstrate its utility in the context of clustering, record matching, movie rating and movie genre prediction.**

***Keywords*-Machine learning, Unsupervised learning, Knowledge representation**

I. Introduction

Structured records have a fixed number of attributes per record instance (albeit some of these attributes may be empty) and are pervasive in many diverse fields. Examples include electronic health records with attributes like patient name, address and health history; e-commerce products with product name, description, price, product rating as attributes.

To perform machine learning, it is routine to first convert records into numeric vectors. Several groups have developed methods to embed records into dense vectors. Examples include representing music records [1], e-commerce products [2], [3] and health records [4]. However, these either assume a specific downstream task for which there is sufficient labeled data or a particular record structure, and cannot be generally applied to different structured records.

Here we discuss Record2Vec (Figure 1), a fast and general encoder framework for generating embeddings of structured records using only unlabeled data. The premise behind our framework is the following: attributes within a record instance are likely to be associated with each other. For example, if a product's "title" is "iPhone 8, 4.7″, 64 GB, Black", then we are likely to see "Apple" in the attribute "brand". Our goal is therefore to learn such attribute associations to build record embeddings. Because the embeddings are trained using information within the structured record itself, we do not need any labeled data for training. This also means that Record2Vec is general, and should be useful for a variety of downstream tasks. Record2Vec is additionally able to handle all types of attributes: descriptive text, categorical, ordinal, single word/token and even numerical. Because the building blocks are based on general token embeddings, it is very fast during inference and is easily scalable.

We demonstrate the utility and generality of our framework on a variety of datasets and supervised/unsupervised tasks. To the best of our knowledge, this is the first description of a general unsupervised representation learning framework for encoding structured records that is applicable across domains. Therefore we make use of TF-IDF representations of records as our baseline for comparison. For consistency, we performed truncated singular value decomposition (SVD) of the sparse TF-IDF representations to obtain dense vectors with the same dimensions as Record2Vec embeddings. Throughout the text, we will refer to this baseline representation as TF-IDF-SVD.

II. Text documents versus structured records

We first draw analogies between structured records and text documents, since our encoder framework is built on principles developed in text and document representation, although we exploit key properties of structured records.

A. Similarities

Text documents are comprised of sentences that are made up of words, so that we can consider a {Documents, Sentence, Words} hierarchy. Similarly, structured records have a natural hierarchy to them: {Records, Attributes, Tokens}.

B. Key differences

- Documents versus Records:
  1) A document can have any number of sentences while there are fixed number of attributes per structured record in a particular dataset.
  2) A document is comprised of sentences that have an inherent sequence structure. Attributes within a structured record are related to each other, but usually have no meaningful ordering.
- Sentences versus Attributes:

1) There are no relationships between sentences across documents. Every attribute is a particular feature descriptor of a record and hence there are strong relationships across record instances.
2) Sentences have a well-defined sequence structure to them. Some attributes of a structured record are ordinal or categorical and therefore their attribute values have no meaningful sequence.
3) Some attributes of a record instance may be empty; there are no empty sentences.

- Words versus Tokens: Words are the building blocks of a language, but there are no restrictions for defining a token used in a structured record.

We leverage these properties of attributes in structured records to build an embedding model that extends from frameworks used for training word embeddings. Because attribute values tend to be associated, we set up a Word2Vec-like model [5], [6], to train *attribute* relations.

III. Proposed Unsupervised Learning Framework

The basic building blocks of Record2Vec are generic string tokens. Here we tokenized by whitespace, but have also experimented with character trigram tokenization with success (data not shown). The objective is to train embeddings for all tokens (vocabulary of V_{tokens} with dimensions D_{em}) from which a record embedding can be generated. To reduce complexity, we ignored token ordering within attributes: each record's attribute is represented by a simple average of the attribute's token embeddings. Furthermore, ordering is only defined if there are multiple tokens present in the attribute value; in general, a structured record can have attributes values consisting of just single tokens (e.g. categorical or ordinal attributes). Simple averaging also helps to maintain the generality of the embedding [7]. Exploring other averaging schemes is left for future work.

A. Negative samples and the triplet loss function

For each attribute embedding from record instance r (i.e. $\mathbf{e}_i^{(r)}$ for the ith attribute), we estimated its negative samples by randomly selecting (without replacement) from the pool of corresponding attribute embeddings across records ($\mathbf{e}_i^{(\neg r)}$). We additionally constrained that $\mathbf{e}_i^{(r)} \neq \mathbf{e}_i^{(\neg r)}$. Then, we made use of the triplet loss function [8] to ensure that we are training token embeddings that identify the correct context (i.e. high similarity to the target), rather than the negative sample by margin α. Every training instance is a triplet: {context attributes, positive target, negative sample}.

B. Continuous Bag of Words (Attributes) Model

The intuition behind the CBoW model is as follows: if we randomly omit an attribute describing a record instance, we may be able to estimate its value from all the remaining attributes. We sequentially omit one attribute from each record and take the sum of all remaining attribute representations to form our anchor. Our loss function that we want to minimize for all records (N_{rec}), attributes (N_{attr}) and negative samples (N_{neg}) is therefore:

$$Loss = \sum_{r=1}^{N_{rec}} \sum_{i=1}^{N_{attr}} \sum_{n=1}^{N_{neg}} max(0, d(\mathbf{anchor}_i^{(r)}, \mathbf{e}_i^{(r)}, \mathbf{e}_i^{(r_n \neq r)}))$$

where

$$\mathbf{anchor}_i^{(r)} = \sum_{j \neq i}^{N_{attr}} \frac{1}{N_t} \sum_{t \in attr_j^{(r)}}^{N_t} \mathbf{TokenEm}[t]$$

and $d(\mathbf{a}, \mathbf{p}, \mathbf{n}) = cosineDist(\mathbf{a}, \mathbf{p}) - cosineDist(\mathbf{a}, \mathbf{n}) + \alpha$.

Here, we set the margin $\alpha = 0.5$ and $D_{em} = 50$. Every model was trained up to 50 epochs with the Adam optimizer [9]. To keep the token vocabulary tractable, we trained each token independent of its use across attributes and records. We further pruned the token vocabulary by removing stopwords, and omitted infrequent words. All infrequent words were represented by a common token for which a generic embedding is trained. Token embeddings were first initialized with random numbers drawn from a standard normal distribution, then normalized so that the token embeddings start with unit magnitude. To reduce noise, we only used the first 200 words of long text.

C. Skip-gram Model

In the SG model, our training data consists of sequentially taking pairs of attributes and setting one as the anchor, and the other as the positive target. The rest of the model is identical to the CBoW version.

D. Missing/empty attributes

Missing data or empty attributes could still be informative. For example, an empty "shoe size" attribute likely indicates that the item is not a shoe. To distinguish empty attributes, we included attribute names (with a special tag to distinguish its usage in attribute values) in our token vocabulary, and incorporated them into our set of tokens when representing attributes. Every attribute value therefore has at least the tagged attribute name as a mandatory token. From our experiments, we found that omitting the tagged attribute name led to weaker performance in general.

E. Inference

Each record's attribute is first represented by the average of embeddings for tokens present in the attribute; empty attributes are assigned vectors of zeros. Record embeddings of size $N_{attr} \times D_{em}$ are then obtained by concatenating attribute embeddings. Each attribute occupies specific indices in the record embedding and hence it is unnecessary to include attribute names as we did during training. An approach obtain a dense embedding of size D_{em} is to simply take the global average of all tokens embeddings present

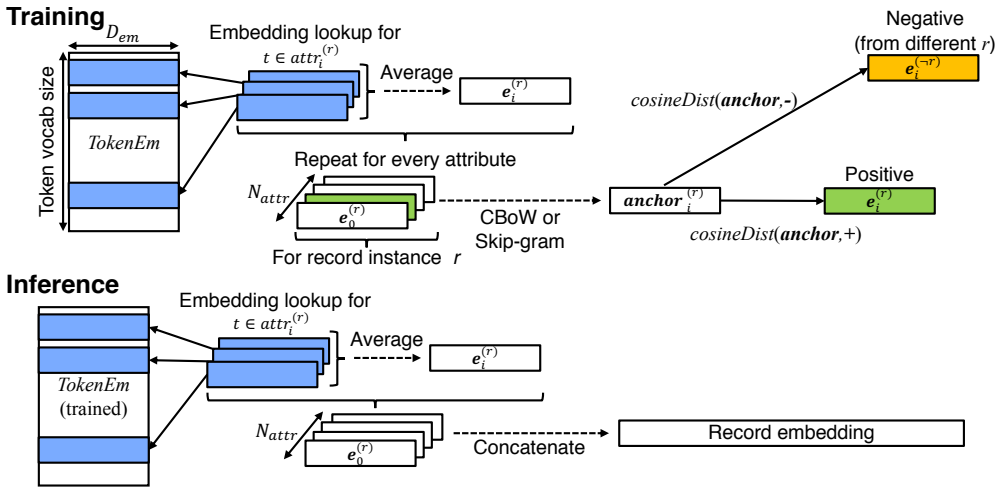

Figure 1. Schematic of Record2Vec as applied to structured records. The ith attribute ($attr_i^{(r)}$) in record instance r is represented by the average of its token (t) embeddings, yielding $\mathbf{e}_i^{(r)}$. After the token embeddings have been trained, in the inference step, every record can be represented by the concatenation of its attribute embeddings.

Table I
OVERVIEW OF DATASETS USED

Dataset	N_{rec}	N_{attr}	V_{tokens}	N_{neg}
Cora	1295	12	559	10
DBLP-ACM	4910	4	2362	10
DBLP-Scholar	66,879	4	15,405	10
Variations	12,596	17	8361	10
Abt-Buy	2173	3	1703	100
Amazon-Google	4589	4	6265	100
IMDb	473,106	6	214,296	10

in the record, regardless of attribute. For clustering, we made use of this dense representation to avoid the curse of dimensionality.

IV. Related Work

A. Representations of relational databases

In some cases, data is present in the form of relational databases, networks or graphs. With relation triplets ({subject, predicate, object}) extracted from these graphs, it is possible to train entity/node (or record) and relation embeddings [10]–[14]. While structured records can be represented as relational graphs, these embedding techniques require a large amount of relations and seem to be less effective in cases with large numbers of records and few relation types [10]. Methods that require traversing the relational graph are in general inappropriate for use on structured records due to the small number of relations *between* record instances, though if there are many relations between record instances, then these aforementioned graph embeddings could be appropriate.

B. Task- or type- specific record embedding

Most record embeddings have been trained for specific downstream tasks. For instance, Prod2Vec [2] and Meta-Prod2Vec [1] were trained for use in recommender systems, and it is unclear if these embeddings are useful for general machine learning tasks. Additionally, the training procedure requires a definition of "surrounding" or "similar" products, which may not be readily available or well-defined. MRNet-Product2Vec [3] uses product titles to predict several other product attributes such as price, weight and size. Our work extends, generalizes and simplifies their approach to be used on any type of record, attribute and input token.

V. Datasets

Statistics of the datasets used in our experiments are summarized in Table I. We only trained token embeddings for tokens that appear at least five times throughout the dataset (regardless of attribute). All numerical attributes were tokenized as strings so that no attribute required special handling. Unless otherwise specified, 50-dimensional TF-IDF-SVD was evaluated for *each* attribute, and then concatenated, so that the dimensions of TF-IDF-SVD are always consistent with Record2Vec.

For supervised learning tasks, we trained both a linear classifier (logistic or linear regression) and a non-linear one (XGBoost [15]). Linear classifiers allow us to more clearly quantify which embedding better captures record structure space, since good quality record embeddings should place similar record instances closer together, making them more linearly separable. On the other hand, non-linear classifiers are more commonly used in practice, and therefore give us insights into the practical utility of Record2Vec. Grid search was used to identify the best hyperparameters for logistic regression and XGBoost. For the latter, we only searched

Table II
SUMMARY OF BLOCKING STATISTICS FOR RECORD MATCHING

	# Record matches	All Pairs	# Blocked	# False negatives
Cora	14,920	839,160	31,645	0
DBLP-ACM	2224	>6 mil	25,509	28
DBLP-Scholar	5347	>168 mil	541,279	33
Abt-Buy	1097	>1.1 mil	147,350	0
Amazon-Google	1300	>4.3 mil	390,239	4

over 100-500 estimators (inclusive) in steps of 100, and with tree depths of 3, 4 and 5. The metrics reported on the test sets throughout are based on classifers trained with the best hyperparameters identified by 3-fold cross-validation. If F1 score is reported, the threshold for classification was determined from a held-out validation set.

A. Scientific publications datasets

1) Cora[1]*:* The Cora dataset consists of hand-labeled scientific publications from the Cora Computer Science Research Paper Engine [16]. 12 attributes were used when training the record embeddings, but for consistency with the other scientific datasets, only four attributes (title, authors, venue and year) were used during record matching.

We used a dense 50-dimensional TF-IDF-SVD for clustering, where global TF-IDF statistics were evaluated for all tokens regardless of which attribute they appeared in. The truncated SVD was then performed on the global TF-IDF matrix. Similarly, for Record2Vec, each record instance is represented by the arithmetic mean of all token embeddings for that record instance, regardless of attribute.

2) DBLP-ACM [17]: The DBLP-ACM dataset has scientific publications from two scientific bibliographic sources: DBLP[2] and ACM digital library[3]. The dataset was designed for record matching tasks, and includes match labels.

3) DBLP-Scholar: The DBLP-Scholar dataset is similar to the DBLP-ACM dataset, except that records are matched between DBLP and Google Scholar scientific publications.

B. E-commerce datasets

1) Variations: The Variations dataset consists of human-labeled e-commerce catalog products. Pairs of products are labeled based on whether they are variations of each other: examples of variations include identical products that vary only in color or size. For clustering, we used dense 50-dimensional vectors, as in the Cora dataset.

2) Abt-Buy [17]: Products were described using name, description and price. For consistency with the Abt.com records, we merged the "manufacturer" and "description" attributes in the Buy.com dataset. All three features were used for Record2Vec training. Since numerical price attribute is not meaningfully represented by TF-IDF-SVD, only product name and description were used for record matching.

3) Amazon-Google [18]: Amazon products were matched with Google search results based on their Universal Product Code. Four attributes (name, description, manufacturer and price) were used in training Record2Vec, but as before, price was omitted during record matching.

Due to the large number of possible pairwise comparisons for record matching, we filtered pairwise comparisons based on a simple trigram matching blocking strategy similar to methods previously adopted [17], [18]. Table II summarizes the statistics of our datasets. The blocked datasets were then split into training/validation/testing (60/20/20).

C. IMDb movie dataset

The IMDb dataset[4] was pre-processed to remove non-movie titles. For training token embeddings, we only used title, directors, writers, principal cast, runtime and start year as attributes. We represented each movie with a 202-dimension vector: a concatenated vector with 50 dimensions representing title, directors, writers, principal cast and 1-dimensional vectors representing numerical attributes (runtime and year). Missing numerical attributes were assigned values of -1. Both the movie embedding and TF-IDF-SVD representations were trained on the full movie dataset; only a subset were used for the supervised learning tasks.

For rating and genre prediction, we further pruned data and removed those with missing rating and/or genre information. This gave a total of 198,188 labeled movies. For rating prediction (where rating is in the interval [0.0,10.0] and represents IMDb's weighted mean of fan ratings), the data was split into training and testing sets (80:20). Because a movie can have multiple genre labels, we trained a classifier for each genre. The data was split into training, validation and testing sets (60:20:20).

VI. RESULTS

A. Embedding captures structure space of records

Good record embeddings should adequately capture the structure space of records and naturally cluster similar records. To quantify this, we performed 100 independent k-means clustering run on both the Cora and Variations datasets, and measured the cluster homogeneity scores of each run. For a clean comparison, we set k to the true number of clusters of each dataset, so that any deviation from a cluster homogeneity score of 1 can be attributed directly to the embeddings. From Figure 2, we observe that both the CBoW and SG versions of Record2Vec significantly outperform TF-IDF-SVD for clustering ($p<0.01$ based on t-test), providing quantitative evidence that Record2Vec embeddings are meaningful dense representations of records.

[1]https://alchemy.cs.washington.edu/data/cora/
[2]http://dblp.uni-trier.de/
[3]http://dl.acm.org/
[4]https://datasets.imdbws.com/

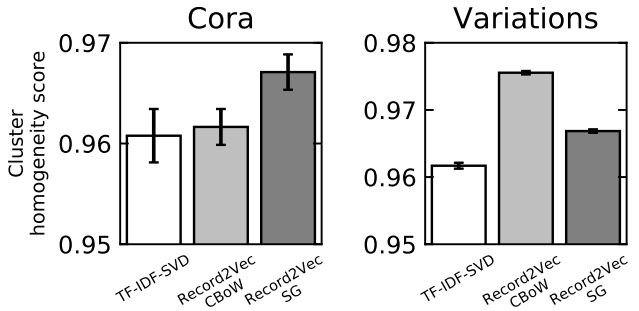

Figure 2. Average cluster homogeneity scores of 100 randomly initialized k-means runs using different embeddings for the Cora (left) and Variations (right) datasets. k was set to the true number of clusters. Standard deviations across the runs are shown as errorbars.

Table III
RECORD MATCHING RESULTS

		Cora	**DBLP -ACM**	**DBLP -Scholar**	**ABT -Buy**	**Amazon -Google**
	SOTA	PRAUC 99.5	F1 97.6	F1 89.4	F1 71.3	F1 62.2
Linear	TF-IDF-SVD	98.39	96.37	78.61	24.95	18.56
	R2V-CBoW	**98.89**	**96.51**	**83.97**	**31.67**	**50.66**
	R2V-SG	98.59	95.93	81.94	27.84	48.29
XGB	TF-IDF-SVD	98.77	**98.01**	84.17	24.88	28.38
	R2V-CBoW	**99.08**	**98.01**	**86.04**	**27.94**	**48.95**
	R2V-SG	98.53	97.42	85.52	24.21	47.90

B. Record matching task

Record matching is the process of matching records from separate sources or of finding duplicate records within a single source. In most cases, token-based string matching is used to determine if two records are identical [19]. Here, we use the record matching task to help us compare the quality of dense TF-IDF-SVD and Record2Vec embeddings.

Consistent with methods in this field, we consider similarity between attributes separately, where each attribute is represented by a 50-dimensional vector derived either from our trained token embeddings or TF-IDF-SVD representation. Then, for each pair of records, we sequentially compared their attributes with the attribute-level cosine and euclidean distances. That is, a $2 \times N_{\text{attributes}}$-dimensional feature vector was used to represent every pairwise record comparison. These features were used in the classifier.

The results for the scientific reference and e-commerce datasets are shown in Table III. State-of-the-art (SOTA; from [10], [18]) benchmark results are also shown for reference. Record2Vec-CBoW consistently outperforms TF-IDF-SVD and Record2Vec-SG for all datasets, regardless of classifier. However, we do not usually attain state-of-the-art record matching capabilities as direct string comparison of attributes are critical features that we have intentionally omitted for a clean comparison between Record2Vec and the TF-IDF-SVD baseline. Nonetheless, observations from the record matching task suggest that Record2Vec embeddings are richer in information than TF-IDF-SVD.

Table IV
PREDICTING MOVIE RATINGS IN THE IMDB DATASET

	Model	Mean squared error
Linear	TF-IDF-SVD	1.717
	Record2Vec-CBoW	**1.638**
	Record2Vec-SG	1.640
XGB	TF-IDF-SVD	**1.338**
	Record2Vec-CBoW	1.401
	Record2Vec-SG	1.410

C. Movie rating and genre prediction

From Table IV and V, we see that Record2Vec outperforms TF-IDF-SVD when a linear classifier is used for both ratings and genre prediction, particularly for less common genres. However, the benefits of using Record2Vec embeddings are negated when XGBoost is used. The main advantage of Record2Vec is that it trains associations between attributes of a record instance, regardless of downstream task. When using XGBoost, we allow the classifier to learn analogous attribute-level associations in a *task-specific* manner. These task-specific attribute associations lead to overall better performance in rating and genre prediction, although Record2Vec still gives competitive results.

VII. CONCLUSIONS

Record2Vec is a generic encoder for generating dense embeddings of structured records for use in various downstream machine learning tasks. We quantitatively compared the performance of Record2Vec in clustering, record matching, classification and regression. In all cases, the dense Record2Vec representations are useful, with advantages over TF-IDF-SVD most pronounced when using linear classifiers.

With precomputed token embeddings, it is straightforward to infer embeddings on new incoming records without additional training, making Record2Vec very practical to deploy. Where necessary, additional types of information such as image embeddings can be concatenated with Record2Vec for a more holistic record representation.

We are currently exploring other more sophisticated approaches to include record embedding directly into the modeling framework, so that we can train both the record and token representations *simultaneously*, analogous to Doc2VecC [20], where word and document embeddings are trained in a single framework . Another direction for improvement would be to incorporate token ordering information when training token embeddings, at least for text-based attributes. If there is sufficient labeled data, Record2Vec can be modified and set up as a semi-supervised learning task.

Table V
PREDICTING MOVIE GENRES IN THE IMDB DATASET

	True positives	Drama 90,982	Comedy 53,862	Documentary 27,455	Romance 23,295	Action 21,356	Crime 17,781	Thriller 16,586	Horror 14,098	Adventure 12,472	Other 67,503
Linear	TF-IDF-SVD	63.54	44.08	39.92	27.47	30.27	20.76	19.59	15.14	15.98	46.32
	Record2Vec-CBoW	**64.71**	45.19	**48.31**	30.58	**40.07**	24.59	**24.22**	32.08	**22.07**	47.14
	Record2Vec-SG	64.64	**45.37**	47.67	**30.59**	39.55	**25.59**	23.46	**32.67**	21.75	**47.59**
XGB	TF-IDF-SVD	**69.25**	**48.31**	**70.87**	**32.70**	**46.93**	**27.35**	**27.42**	**43.29**	**31.12**	**49.49**
	Record2Vec-CBoW	67.07	47.61	68.26	32.28	45.19	26.84	27.04	39.85	23.79	48.57
	Record2Vec-SG	67.16	47.27	67.95	31.73	44.75	27.00	26.69	39.80	22.06	48.67

ACKNOWLEDGMENT

We thank the Amazon AI Lab for useful discussions and feedback.

REFERENCES

[1] F. Vasile, E. Smirnova, and A. Conneau, "Meta-prod2vec: Product embeddings using side-information for recommendation," in *Proceedings of the 10th ACM Conference on Recommender Systems*, ser. RecSys '16, 2016, pp. 225–232.

[2] M. Grbovic, V. Radosavljevic, N. Djuric, N. Bhamidipati, J. Savla, V. Bhagwan, and D. Sharp, "E-commerce in your inbox: Product recommendations at scale," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '15, 2015, pp. 1809–1818.

[3] A. Biswas, M. Bhutani, and S. Sanyal, "Mrnet-product2vec: A multi-task recurrent neural network for product embeddings," in *Machine Learning and Knowledge Discovery in Databases*, Y. Altun, K. Das, T. Mielikäinen, D. Malerba, J. Stefanowski, J. Read, M. Žitnik, M. Ceci, and S. Džeroski, Eds. Cham: Springer International Publishing, 2017, pp. 153–165.

[4] R. Miotto, L. Li, B. A. Kidd, and J. T. Dudley, "Deep patient: An unsupervised representation to predict the future of patients from the electronic health records," *Scientific Reports*, vol. 6, p. 26094, May 2016, article.

[5] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS'13, 2013, pp. 3111–3119.

[6] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *CoRR*, vol. abs/1301.3781, 2013.

[7] J. Wieting, M. Bansal, K. Gimpel, and K. Livescu, "Towards universal paraphrastic sentence embeddings," *International Conference on Learning Representations*, 2016.

[8] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 815–823.

[9] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014.

[10] A. Bordes, X. Glorot, J. Weston, and Y. Bengio, "A semantic matching energy function for learning with multi-relational data," *Machine Learning*, vol. 94, no. 2, pp. 233–259, Feb 2014.

[11] M. Nickel, V. Tresp, and H.-P. Kriegel, "A three-way model for collective learning on multi-relational data," in *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ser. ICML'11, 2011, pp. 809–816.

[12] B. Yang, W. tau Yih, X. He, J. Gao, and L. Deng, "Embedding entities and relations for learning and inference in knowledge bases," *International Conference on Learning Representations*, 2015.

[13] P. Ristoski and H. Paulheim, "Rdf2vec: Rdf graph embeddings for data mining," in *The Semantic Web – ISWC 2016*, P. Groth, E. Simperl, A. Gray, M. Sabou, M. Krötzsch, F. Lecue, F. Flöck, and Y. Gil, Eds. Cham: Springer International Publishing, 2016, pp. 498–514.

[14] A. Grover and J. Leskovec, "Node2vec: Scalable feature learning for networks," in *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16, 2016, pp. 855–864.

[15] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16, 2016, pp. 785–794.

[16] P. Singla and P. Domingos, "Entity resolution with markov logic," in *Proceedings of the Sixth International Conference on Data Mining*, ser. ICDM '06, 2006, pp. 572–582.

[17] H. Kopcke, A. Thor, and E. Rahm, "Learning-based approaches for matching web data entities," *IEEE Internet Computing*, vol. 14, no. 4, pp. 23–31, 2010.

[18] H. Köpcke, A. Thor, and E. Rahm, "Evaluation of entity resolution approaches on real-world match problems," *Proc. VLDB Endow.*, vol. 3, no. 1-2, pp. 484–493, Sep. 2010.

[19] A. K. Elmagarmid, P. G. Ipeirotis, and V. S. Verykios, "Duplicate record detection: A survey," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 1, pp. 1–16, Jan 2007.

[20] M. Chen, "Efficient vector representation for documents through corruption," *5th International Conference on Learning Representations*, 2017.