Fusing Multi-label Classification and Semantic Tagging

Jörg Kindermann[1,2] and Katharina Beckh[1,2]

[1] Fraunhofer IAIS, Sankt Augustin, Germany
[2] Competence Center for Machine Learning Rhine-Ruhr
{joerg.kindermann,katharina.beckh}@iais.fraunhofer.de

Abstract. Companies have an increasing demand for enriching documents with metadata. In an applied setting, we present a three-part workflow for the combination of multi-label classification and semantic tagging using a collection of key-phrases. The workflow is illustrated on the basis of patent abstracts with the CPC scheme. The key-phrases are drawn from a training set collection of documents without manual interaction. The union of CPC labels and key-phrases provides a label set on which a multi-label classifier model is generated by supervised training. We show learning curves for both key-phrases and classification categories, and a semantic graph generated from cosine similarities. We conclude that, given sufficient training data, the number of label categories is highly scalable.

Keywords: multi-label classification · semantic tagging · prediction-based embedding spaces · patents.

1 Introduction

For strategic developments, businesses and research organizations have an interest in identifying competences or trends in their respective organization and in comparison to competing institutions. Extracting this information manually among heterogeneous data is time-consuming which is partly complicated by different underlying classification schemes, e.g. from patents or publications. Therefore, there is an increasing demand for metadata [8] that combines categories from classification schemes with semantic tags.

The automatic single-label classification of documents is well-researched [21] [1] while multi-label classification with large numbers of labels still is a challenge [16]. The combination of classification and semantic tagging is also less explored. Advances in the distributed representation of words have provided the necessary basis for this combination [14] and recent work allows to achieve both steps together in a document processing workflow [18].

To tackle the fusion of classification and semantic tagging in an applied setting, we introduce a basis workflow which allows to classify and tag documents

at once. For that we start by introducing the tools, namely the model, data and evaluation metrics (Section 3). Subsequently, we put the approach into context by describing a use case within the Fraunhofer society that aims to extract information from existing data sources (Section 4.1). As patent data is an important base for innovation research and because it exhibits one of the largest and prominent classification schemes, we employ it to demonstrate the workings of our approach.

Following the use case, we describe the three-part workflow in detail (Section 4.2). A set of key-phrases is collected in an unsupervised procedure from a training set of documents. The union of category labels and key-phrases provides a label set on which a multi-label classifier model is trained. Following the model training, we furthermore describe how to extract embedding vectors to visually represent classification categories and key-phrases together in a semantic graph. We depict learning curves with appropriate metrics and a cutout of the semantic graph. We conclude that the workflow scales to a larger amount of documents and can be applied on documents in various domains.

2 Related Work

Multi-label classification with a large number of categories has been notoriously difficult. A first break-through that made classification of texts possible without relying on manually designed features was the Support Vector Machine [5], [10]. However, the computational effort grows considerably with the number of labels, making the training of classification problems with thousands of labels intractable. Semantic tagging, i.e. the assignment of key-phrases to a text, in an unsupervised way was achieved by applications of the Latent Dirichlet Allocation topic model [3].

Both steps, multi-label classification and semantic tagging, in a document processing workflow could recently be combined with the advent of the *StarSpace* algorithm [18] based on embedding vector spaces. This algorithm implements the concept of prediction-based embedding spaces.

Since Elman's seminal paper [7] on recurrent neural networks and their training on sequences, in particular sentences as sequences of words, there have been many efforts to improve the storage capacity and reduce the computational complexity of such systems. The *Word2Vec* algorithms [14] were a path-breaking invention in this direction which for the first time made it possible to represent semantic properties of words derived from their actual usage in large quantities of texts. This algorithm exceeded capacities of systems known so far by orders of magnitude. Levy and Goldberg [12] showed that the *Word2Vec* algorithms are closely related to counting-based vector representations by matrix-factorization mappings. An example is a vector-space based on *PMI* (point-wise mutual information) values. This finding supports confidence in the semantic properties of prediction-based embedding spaces, such as the StarSpace model, which are explored by cosine similarity. This is due to their close relationship to PMI-based

representations. Important follow-up developments of Word2Vec were Glove [15] and FastText [4].

Recent applications of StarSpace have been published in the areas of ontologies [9] and knowledge graphs [20] that are related to our use case. Regarding other recent work, transformer-based architectures [6] are also suitable for multi-label classification.

3 Methods

3.1 StarSpace

We chose StarSpace [18], a general-purpose neural embedding model which can be used for multi-label classification and tagging. It is based on a *bag of entities* representation. *Entities* can be texts, labels, meta-data like authors, source URLs etc. Starspace thus is capable of learning relations between items of various types and origins. The bag of entities representation is a high dimensional vector in an embedding space which may include labels. The actual learning algorithm is a stochastic gradient descent optimization of a special loss function

$$\sum_{(a,b)\in E^+, b^-\in E^-} L^{batch}(sim(a,b), sim(a,b_1^-), ..., sim(a,b_k^-)) \tag{1}$$

where entities a and b are drawn from the set E^+ of positive examples, and entities b^- are drawn from the set E^- of negative examples. In our use case (section 4.1) the entities are the patent abstracts and their labels and key-phrases. The k-negative sampling strategy of [14] is used. The similarity function can be chosen from $\{cosine, dot product\}$. The loss function L_{batch} has two implementations:

- margin ranking loss: $\max(0, \mu - sim(a,b))$ with margin parameter μ
- the negative log loss of the softmax function: $-\log(\frac{e^{y_i}}{\sum_j e^{y_j}})$

During the optimization run, the similarity function $sim(\Delta, \Delta)$ is "learned". It can subsequently be used to measure the similarity between *entities*. For classification, a label is predicted for a given input a as $\max_{\hat{b}}(sim(a,\hat{b}))$ over the set of possible labels $\hat{b}$. This feature can be used to output a ranking of labels according to their similarity, implementing multi-label classification.

3.2 Data

In our experiments, we employ a sample of patent abstracts from the United States Patent and Trademark Office (USPTO)[3] from the month of January 2020 which amounts to 22.000 abstracts. The classification scheme that we use is the Cooperative Patent Classification (CPC). The CPC hierarchy is illustrated in Fig.1 and consists of *section*, *class*, *subclass*, *maingroup* and *subgroup*.

[3] https://developer.uspto.gov/product/patent-grant-full-text-dataxml

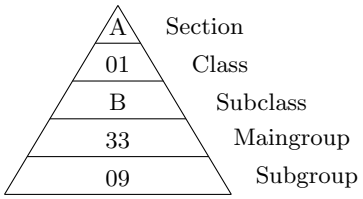


Fig. 1: CPC hierarchy illustrated with an example category

Label	Quantity
Main-CPC	100
Further-CPC	250
Key-phrases	200

Table 1: Number of labels per category

We focus on the first three levels, namely section, class and subclass. The data contains a *Main-CPC* which serves as the main category of the patent and *Further-CPC* categories which are also applicable categories (see Fig. 5(b) for examples). We selected a subset of all possible labels with respect to the number of examples available in our data collection. Table 1 shows the numbers of selected labels in both categories. For the category *key-phrases* see section 4.2.

3.3 Evaluation Metrics

We evaluated the experiments based on two metrics:

- **F1 value** is the well-known harmonic mean of *precision* and *recall* measures. We used the F1 value to assess the performance on the Main-CPC labels, because it is suited to evaluate single-label classification tasks mainly.
- **Coverage-rank** [19] with a real-valued ranking function $f(.,.)$

$$\text{coverage}(f) = \frac{1}{p}\sum_{i}\max_{y\in Y}\text{rank}_f(x_i, y) - 1 \tag{2}$$

  counts how many steps have to be taken to move down the ranked label list to cover all the relevant labels of the example. The coverage-rank was used to assess the performance on the Further-CPC labels and key-phrases. It seems to be more adequate to multi-label classification than the F1 value. Another important reason is that we want to train the model on a semantic tagging task, which would be thwarted by an exclusive optimization according to F1 values. The reason is that semantic tagging is expected to tag documents with a certain key-phrase that is not literally contained in the document but is nevertheless highly relevant to the document content and topic. This desired behavior would, however, result in a degraded F1 value because it would be counted as a false positive.

4 Experiments

4.1 Use Case

Here, we first describe the applied benefit of our approach in the context of a current project. Within the project "Fraunhofer Digital" a data hub has been

created which will cover a variety of datasets, ranging from publications and patents to project descriptions. All the datasets contain valuable information about the competence landscape and, in particular, patent data is important for the strategic technology and innovation management within Fraunhofer.

One key challenge is that patents are only mapped to a patent classification system. There is no basis in linking the classification to information outside of the scheme. In this use case it is desired to find similarities between patents and at a glance we want to identify the most suitable key-phrases. This makes it for example easier to determine current technologies and technology trends.

Our approach is to extract and assign information inherent in the patents that exceeds the common patent classification. We achieve this by employing key-phrase extraction. By providing key-phrases on top of the classification, the model provides comprehensible information for readers and therefore serves as a base to facilitate work for employees. In the "Fraunhofer Digital" use case we apply this approach also to publication data using more data to create several classification models. For this paper, we narrow our focus to patent samples. In the following, we describe the workflow in more detail.

4.2 Workflows

Key-phrase Extraction. We collect a list of key-phrases from the pool of training documents using the *RAKE* (Rapid Automatic Keyword Extraction) algorithm [17]. We chose RAKE, because it does not depend on sophisticated preprocessing operations as named-entity recognition and training of neural networks as in [13]. RAKE operates in an unsupervised manner on individual documents. It identifies key-phrases by extracting phrases between stopwords (e.g. "the", "a") and by analyzing the frequency of word appearance and word co-occurrence.

Because RAKE works on single documents, the frequent extraction of non-informative standard key-phrases like section headings ("Related Work", etc.) is expected. It can be avoided by detecting and elimiating those phrases based on an information-theoretic measure like TF-IDF (Term Frequency - Inverse Document Frequency) [2] or Importance Weight [11]: We chose TF-IDF and keep only those phrases which contain at least one term with a value above a certain threshold (to be set as a hyper-parameter). The resulting list usually is still too large. Therefore, we select the n most frequent phrases. In the experiment described here, we chose 200 key-phrases (see Table 1). Examples from this set of key-phrases are "search engine" or "application programming interface" and more are depicted in Fig. 5. The selected key-phrases define the gold-standard for F1 value optimization.

Model Training. The key-phrases together with the Main-CPC and Further-CPC labels define the set of StarSpace labels to be trained (see Fig. 5(b) for examples). Taking the abstracts and the labels, the StarSpace model is trained (Fig. 2 top) with a pre-determined number of iterations on the training set. From

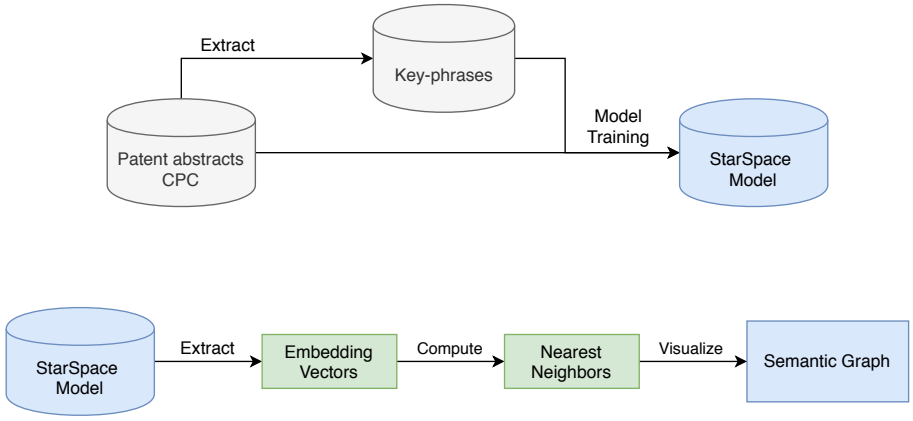


Fig. 2: The training workflows. The top shows key-phrase extraction and the bottom illustrates the construction of a semantic graph

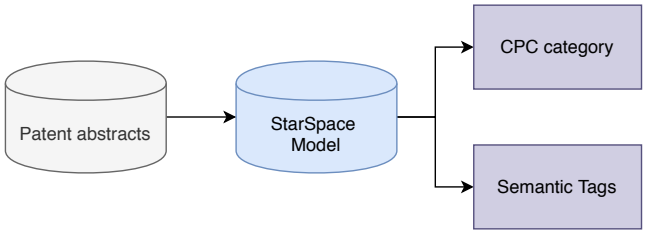


Fig. 3: The prediction workflow. Patent abstracts are fed into the StarSpace model which computes CPC categories and tags

the trained model we export the embedding vectors of the labels and construct a semantic graph that represents the cosine-similarity based k-nearest-neighbor relations of the labels (Fig. 2 bottom). This graph serves as a human-readable quality reference of the model. It is not directly used for the prediction workflow.

To optimize hyper-parameters we used a fixed training dataset of ∼13.000 documents and a test set of ∼8.800 documents (60%/40% split). We evaluated model performances for the CPC scheme from level 1 *Section* to level 4 *Main-group* (see Fig. 1). Results are reported exclusively for level 3 *Subclass*, because this was the most detailed level for which we could achieve satisfactory results.

The StarSpace algorithm has several hyper-parameters[4] which need to be explored in separate evaluations. We optimized 9 of them (see Table 2).

StarSpace param.	Description	Explanation
iterations	number of training iterations	an iteration includes n minibatches
minCount	min frequency of terms	less frequent terms are eliminated
ngrams	ngrams of terms	ngrams up to n terms
dim	embedding dimension	the dimension of embeding vectors
lr	learning rate	learning rates are set to $<= 0.05$
batchSize	batch size	number of items in a minibatch
loss	loss function	the functions *hinge* (i.e. margin ranking) or *softmax*
similarity	similarity measure	*cosine similarity* or *dot product* of embedding vectors
adagrad	stochastic gradient optimizer	adagrad can be switched on or off

Table 2: Description of hyperparameters that we optimized

Model Prediction. New documents (without CPC-label) are assigned their CPC-labels and key-phrases by the trained StarSpace model (see Fig. 3). For each test document the model outputs a weight for each of the labels. Therefore, we need another hyper-parameter *weight-threshold* to cut-off the list of output labels sorted decreasingly by weight to achieve adequate F1 values.

4.3 Results

Attainable Model Performance Figure 4 shows a typical development of F1 and coverage-rank values during a training run of 640 iterations, a weight-

[4] see https://github.com/facebookresearch/StarSpace

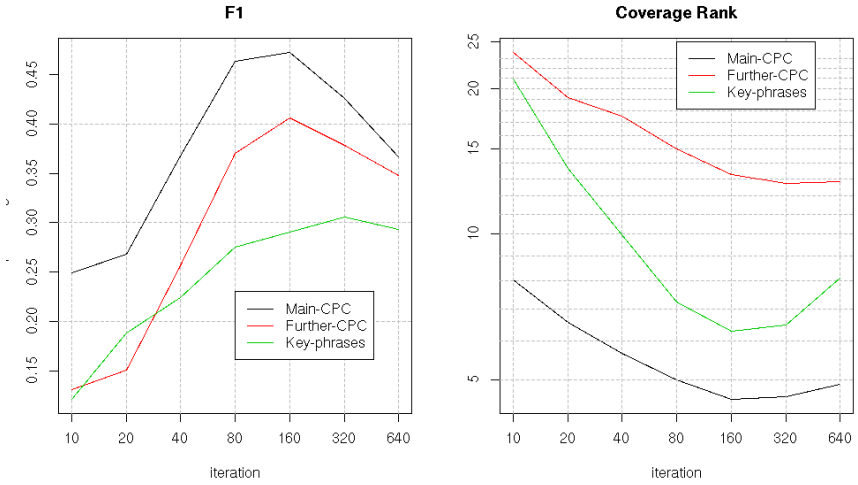


Fig. 4: Example illustration of learning curves of F1 value and coverage-rank for Main-CPC (black), Further-CPC (red), and key-phrase (green) labels.

threshold of 0.35 and otherwise optimal StarSpace parameters. We see that optimal values of F1 and coverage-rank occur in the same range of iterations. Note that large F1 values but small coverage-rank values are better. The overall F1 values are not very competitive. This is partly due to the limited number of documents we use. Moreover, optimizing the F1 value is only a secondary goal. It only makes sense for the Main-CPC values, because they are single-label categories. For the Further-CPC labels and a fortiori for the key-phrases we cannot define the F1 measure in a fully consistent way. This would require a predefined ordering on the multi-label categories which is not given. After all, the behavior of the different label sets is as expected: the single-label Main-CPC categories show better performance with respect to F1 compared to the multi-label categories Further-CPC and key-phrases.

The more important evaluation criterion is the coverage-rank, because it gives an estimate on the precision of the output of non-sorted multi-labels. Here we see the Main-CPC labels again performing best, as expected. The second-best performance of key-phrases and the rather large distance of the Further-CPC values to the other two cases is not expected and needs an explanation: All Further-CPC labels are drawn from the same category system as the Main-CPC labels. The most relevant of them is the Main-CPC label, and all others are Further-CPC labels. The sequence of CPC categories may thus be different for thematically closely related patent abstracts and result in different Main/Further-CPC label sets. This seems to be more difficult to learn for a model than categorizations from disjoint label sets. The fact that we have more Further-CPC labels than keywords may also add to the performance differences.

Semantic Tagging A trained StarSpace model contains exportable embedding vectors for both the terms occurring in the training documents **and** all category labels. This allows to define a k-nearest-neighbor relation on the labels with the cosine-similarity of their embedding vectors. A similar relation exists between the label embeddings and document texts based on the *bag-of-ngrams* representation of the documents[5]. This allows to assign k-nearest-neighbor key-phrase labels as semantic tags to documents. It is difficult to rate the appropriateness of such tagging directly. We therefore display a k-nearest-neighbor graph of labels from all three categories in Fig. 5.

This sub-graph is centered around the Main-CPC level 3 category "G06F - electric digital data processing" and shows the neighboring color-coded Main-CPC (red), Further-CPC (light blue) and key-phrase (cyan) labels[6]. The complete graph contains all 550 labels as nodes. The directed edges in the graph code the cosine similarity between the label embeddings. More similar labels are connected by stronger edges. Note that the linear distance of labels in this graph therefore is **not** an indicator of their embedding similarity. The edge color is set by its source label. In particular, we can observe that the Main-CPC labels and the Further-CPC labels of identical categories (for example G06F) are connected strongly vice-versa, as one would expect.

Semantic tagging now works as follows: if a document is classified, for example as M_G06F, it gets assigned the Further-CPC labels G06F and H04L, as well as the key-phrases "search engine", "client system", "operating system", "computer processor" and possibly more key-phrases that are not displayed in this graph cutout. This tagging behavior is a major difference from other tagging algorithms in that it may assign key-phrases to a document that are **not** contained in the document itself.

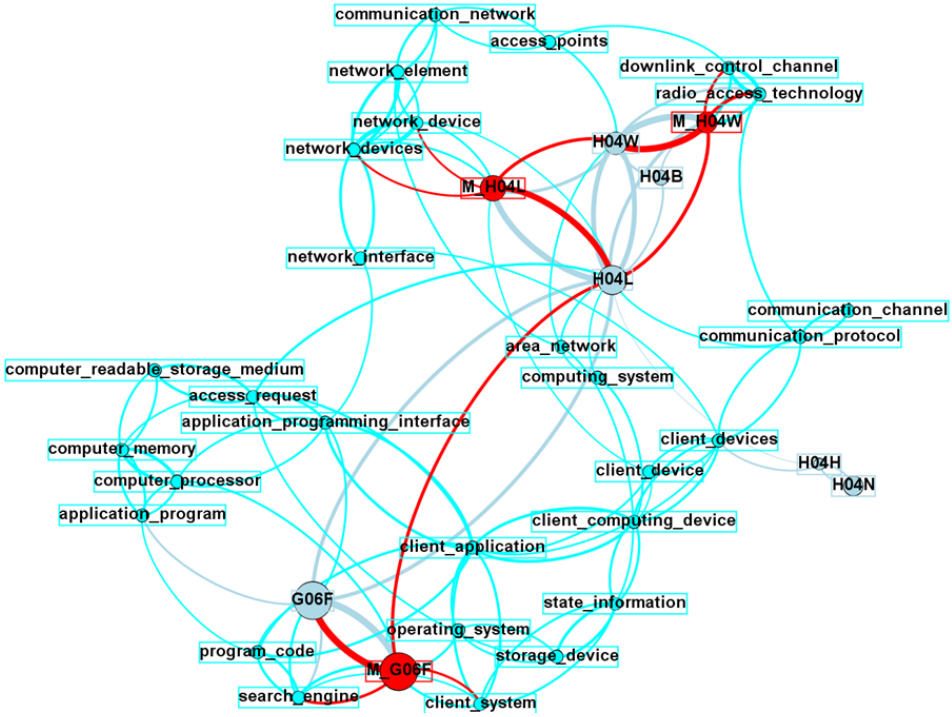
4.4 Limitations and Recommendations

The classification and tagging workflow presented here has some intrinsic limitations which we will shortly discuss in this section.

- **Specificity of key-phrases**: We advise to investigate the specificity of the key-phrases that are extracted by the RAKE algorithm followed by TF-IDF filtering. Depending on the particular properties of a training collection, many of the key-phrases may occur in a large number of multi-label categories. It is up to the experimenter to create a mix of more frequent and more specific key-phrases if required.
- **Number of labels**: Though scalable in a large range there surely exist upper limits of the number of labels in a multi-label classification regime. These limits are related to the number of documents in the training set, but also to the skewedness of label distributions. We did not run quantitative investigations on this topic but from our general experience with StarSpace

[5] For details see https://github.com/facebookresearch/StarSpace

[6] For details see https://www.cooperativepatentclassification.org/cpcSchemeAndDefinitions/table



(a)

CPC category	Description
G06F	Electric digital data processing
H04	Electric communication technique
H04B	Transmission
H04H	Broadcast communication
H04L	Transmission of digital information
H04N	Pictorial communication
H04W	Wireless communication networks

(b)

Fig. 5: (a) Semantic graph generated from cosine similarities of labels and key-phrases. Main-CPC is illustrated in red, Further-CPC in light blue and key-phrases in cyan. (b) The CPC categories and their description.

models in several domains we would state the following: The number of labels should not exceed 1-2% of the number of training data, and with respect to skewedness of distribution the frequency ratio of the least frequent and the most frequent label should not exceed 0.01. One way to circumvent the limit of label numbers would be to split labels into subsets and train several StarSpace models, one on each subset. Doing this, one has to take into account that the label weights in the model output cannot be compared across models. Therefore it makes sense to define subsets accordingly - for example *category labels*, *frequent key-phrases*, and *specific key-phrases*.
- **Model and processing resources**: StarSpace models can be very large with large numbers of training data and large n for the *ngram* parameter. Model sizes of more than 10GB are common, which also require corresponding RAM sizes to process. The StarSpace program is thread-parallel, but training wall-clock times can nevertheless exceed a day for large training sets and many training iterations. Compared to training times, the prediction time of a single document is small in the range of milliseconds.

5 Conclusion

We presented a detailed three-part workflow that allows to combine multi-label classification with semantic tagging demonstrated on patent abstracts with more than 200 CPC categories. An annotated large training set is needed to accomplish good results. The semantic tagging is based on a set of key-phrases extracted by an unsupervised algorithm from a training set. The predicted key-phrases do not have to occur literally in the tagged document. The number of labels and key-phrases is highly scalable, given sufficient training data.

For future work, we plan to test our approach by replacing StarSpace with a deep neural network architecture. We already performed preliminary experiments with Transformer architectures, i.e. BERT [6], on the patent dataset and also on other textual datasets with different classification systems. The results on the patent dataset suggest that the performance of BERT is significantly worse than StarSpace with this amount of data and tests of both StarSpace and BERT on much larger datasets resulted in equal performance. We are planning to consolidate this hypothesis in more experiments.

Acknowledgements We thank the project team of Fraunhofer Digital for the opportunity, and Sven Giesselbach for helpful comments. This research has been funded by the Federal Ministry of Education and Research of Germany as part of the competence center for machine learning ML2R (01IS18038B).

References

1. Adhikari, A., Ram, A., Tang, R., Lin, J.: Docbert: Bert for document classification. arXiv preprint arXiv:1904.08398 (2019)

2. Aizawa, A.: An information-theoretic perspective of tf–idf measures. Information Processing & Management **39**(1), 45–65 (2003)
3. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. Journal of machine Learning research **3**(Jan), 993–1022 (2003)
4. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics **5**, 135–146 (2017)
5. Cortes, C., Vapnik, V.: Support-vector networks. Machine learning **20**(3), 273–297 (1995)
6. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
7. Elman, J.L.: Finding structure in time. Cognitive science **14**(2), 179–211 (1990)
8. Hirschmeier, S., Schoder, D.: Combining word embeddings with taxonomy information for multi-label document classification. In: Proceedings of the ACM Symposium on Document Engineering 2019. pp. 1–4 (2019)
9. Jiménez-Ruiz, E., Agibetov, A., Chen, J., Samwald, M., Cross, V.: Dividing the ontology alignment task with semantic embeddings and logic-based modules. arXiv preprint arXiv:2003.05370 (2020)
10. Joachims, T.: Svmlight: Support vector machine. SVM-Light Support Vector Machine http://svmlight. joachims. org/, University of Dortmund **19**(4) (1999)
11. Leopold, E., Kindermann, J.: Text categorization with support vector machines. how to represent texts in input space? Machine Learning **46**(1-3), 423–444 (2002)
12. Levy, O., Goldberg, Y.: Neural word embedding as implicit matrix factorization. In: Advances in neural information processing systems. pp. 2177–2185 (2014)
13. Mahata, D., Kuriakose, J., Shah, R., Zimmermann, R.: Key2vec: Automatic ranked keyphrase extraction from scientific articles using phrase embeddings. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers). pp. 634–639 (2018)
14. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems. pp. 3111–3119 (2013)
15. Pennington, J., Socher, R., Manning, C.D.: Glove: Global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). pp. 1532–1543 (2014)
16. Prabhu, Y., Kag, A., Harsola, S., Agrawal, R., Varma, M.: Parabel: Partitioned label trees for extreme classification with application to dynamic search advertising. In: Proceedings of the 2018 World Wide Web Conference. pp. 993–1002 (2018)
17. Rose, S., Engel, D., Cramer, N., Cowley, W.: Automatic keyword extraction from individual documents. Text mining: applications and theory **1**, 1–20 (2010)
18. Wu, L.Y., Fisch, A., Chopra, S., Adams, K., Bordes, A., Weston, J.: Starspace: Embed all the things! In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
19. Zhang, M.L., Zhou, Z.H.: A review on multi-label learning algorithms. IEEE transactions on knowledge and data engineering **26**(8), 1819–1837 (2013)
20. Zhang, Q., Sun, Z., Hu, W., Chen, M., Guo, L., Qu, Y.: Multi-view knowledge graph embedding for entity alignment. arXiv preprint arXiv:1906.02390 (2019)
21. Zhang, X., Zhao, J., LeCun, Y.: Character-level convolutional networks for text classification. In: Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1. pp. 649–657. NIPS'15 (2015)