Exploring Word Embeddings in CRF-based Keyphrase Extraction from Research Papers

Krutarth Patel
kipatel@ksu.edu
Kansas State University
Manhattan, Kansas, United States

Cornelia Caragea
cornelia@uic.edu
University of Illinois at Chicago
Chicago, Illinois, United States

ABSTRACT

Keyphrases associated with research papers provide an effective way to find useful information in the large and growing scholarly digital collections. However, keyphrases are not always provided with the papers, but they need to be extracted from their content. In this paper, we explore keyphrase extraction formulated as sequence labeling and utilize the power of Conditional Random Fields in capturing label dependencies through a transition parameter matrix consisting of the transition probabilities from one label to the neighboring label. We aim at identifying the features that, by themselves or in combination with others, perform well in extracting the descriptive keyphrases for a paper. Specifically, we explore word embeddings as features along with traditional, document-specific features for keyphrase extraction. Our results on five datasets of research papers show that the word embeddings combined with document specific features achieve high performance and outperform strong baselines for this task.

CCS CONCEPTS

• **Information systems → Information extraction**.

KEYWORDS

Keyphrase extraction, sequence labeling, Conditional Random Fields, word embeddings, feature representations

ACM Reference Format:
Krutarth Patel and Cornelia Caragea. 2019. Exploring Word Embeddings in CRF-based Keyphrase Extraction from Research Papers. In *Proceedings of the 10th International Conference on Knowledge Capture (K-CAP '19), November 19–21, 2019, Marina Del Rey, CA, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3360901.3364447

1 INTRODUCTION

Keyphrase extraction is the task of automatically extracting a small set of meaningful words or phrases that can accurately summarize the topics discussed in a document [24]. Keyphrases therefore provide a high-level topic description of a document, can allow for *efficient* data organization and information processing, and have a high impact on document understanding and reading comprehension. Additionally, keyphrases associated with a document are often useful in many applications such as document indexing, classification, clustering, recommendation, and summarization [1, 4, 48, 54], contextual advertising [58], and opinion mining [8]. Due to their high importance, many approaches to keyphrase extraction have been proposed in the literature. Most of these approaches are either supervised or unsupervised and work in two steps. First, a set of candidate words or phrases are formed using certain part-of-speech (POS) tags (e.g., nouns and adjectives) and patterns (e.g., at least one noun possibly preceded by adjectives) [27]. Second, the candidate words or phrases are ranked based on the aggregated "informativeness" scores of the individual words comprising a phrase [45, 55] in unsupervised approaches, or are classified as keyphrases or non-keyphrases based on a set of linguistic and statistical features such as *tf-idf*, POS tags, and the relative position of phrases in documents [20, 27] in supervised approaches.

More recently, researchers started to address keyphrase extraction as a sequence labeling task [9, 22, 59]. For example, Gollapalli et al. [22] formulated keyphrase extraction as sequence labeling and showed on several datasets of research papers that using Conditional Random Fields (CRFs) can improve the performance over previous supervised and unsupervised models for this task. The authors used word features such as "WordIsCapitalized," "WordIsStopword," NP-chunking and POS tags, and "WordIsInTitle," as well as their combinations, e.g., "WordIsInTitle and Word POS tag=noun." However, this approach does not capture the semantics of words in context that are often hidden in text. We posit that incorporating word semantics in context in a CRF model has the potential to further improve the performance of keyphrase extraction from research papers.

One way to capture word semantics is to use word embeddings or the distributed vector representations of words, trained on very large collections of documents in an unsupervised fashion [46]. Along this line, Turian et al. [53] showed on two NLP tasks, chunking and named entity recognition, that using word embeddings in existing supervised systems is a simple and general method to improve their performance. Marujo et al. [42] and Mahata et al. [40] showed promising results on keyword extraction by incorporating word embeddings into supervised and unsupervised models. However, unlike these works, in this paper, we investigate the discriminative power of word embeddings in sequence labeling based models. Specifically, we address keyphrase extraction as sequence labeling using CRF models and explore word embeddings as features, by themselves or in combination with other statistical and linguistic features, to determine their discriminative power in correctly extracting the descriptive keyphrases for a research paper.

K-CAP '19, November 19–21, 2019, Marina Del Rey, CA, USA

ACM ISBN 978-1-4503-7008-0/19/11…$15.00
https://doi.org/10.1145/3360901.3364447

Our contributions are as follows:

- We propose to incorporate word semantics in CRF models for keyphrase extraction through the use of word embeddings. We study the sensitivity of CRFs based on word embedding types, i.e., those pre-trained on Google News as well as those trained on a large collection of ACM research papers. As part of our contributions, we will make available the IDs of our ACM dataset and the word embeddings.[1]
- We contrast the CRFs that use word embeddings with a more sophisticated model, Bi-LSTM-CRF [3]. In Bi-LSTM-CRF, the input and output layers are not directly connected as in CRF, but instead a Bi-LSTM (Bidirectional Long Short Term Memory) layer is inserted between them to exploit the long term dependencies in the text. Through extensive experiments, we show that directly using word embeddings in CRF models is a simple and general method to improve CRFs' performance and that the non-linearity brought by the Bi-LSTM layer yields a representation that is not always useful for the CRF model.
- We experimentally show that the CRF models that use word embeddings in addition to features extracted from the document itself outperform strong baselines and other previous approaches for keyphrase extraction.

The rest of the paper is organized as follows: In the next section, we summarize the related work. We describe our sequence labeling problem and the proposed features in Section 3. In Section 4, we present the datasets used for evaluation. The experimental setup and results are discussed in Section 5, followed by conclusions and future directions of our work in Section 6.

2 RELATED WORK

Keyphrase extraction has been the focus of many supervised and unsupervised studies [24]. In the supervised studies, the prediction is done based on a selection of linguistic and statistical features extracted from the text of a document, e.g., a word or phrase part of speech (POS) tags, *tf-idf* scores, and position information, used in conjunction with machine learning classifiers such as Naïve Bayes and Support Vector Machines [20, 27, 47, 51]. These features were also combined with features extracted from external sources such as WordNet and Wikipedia [38, 43] or from various document neighborhoods, e.g., a document's citation network [11, 12].

Unsupervised studies include phrase scoring methods based on measures such as *tf-idf* and topic proportions [6, 37, 61], graph-based ranking using centrality measures, e.g., PageRank scores [19, 23, 32, 45, 55], and keyphrase selection from topics detected using topic modeling [34, 52]. Blank et al. [10] ranked keyphrases for a target paper using keyphrases from the papers that are cited by the target paper or that cite at least one paper that the target paper cites. In the unsupervised context, several extensions of PageRank have been proposed that make use of a document's citation network [21] or that bias the random walk based on the words' positions in text [19] or the words' topic distribution estimated using topic models [36]. In order to add semantic relatedness between the words in a word graph, Martinez-Romo et al. [41] used information from WordNet. The best performing SemEval 2010 system used term frequency thresholds to filter out phrases that are unlikely to be keyphrases, where the thresholds were estimated from the data [17]. The candidate phrases were ranked using *tf-idf* in conjunction with a boosting factor that was aimed at reducing the bias towards single words. Danesh et al. [16] computed an initial weight for each phrase based on a combination of the *tf-idf* score and the first position of a phrase in a document. Phrases and their initial weights were then incorporated into a graph-based algorithm which produces the final ranking of keyphrases. Adar and Datta [2] extracted keyphrases by mining abbreviations from scientific literature and built a semantic hierarchical keyphrase database. Many of the above approaches, both supervised and unsupervised, are compared and analyzed in the ACL survey on keyphrase extraction by Hasan and Ng [25]. Fan et al. [18] used a random-walk method to extract keyphrases using word embeddings combined with the features of candidate words and edges from the word graph.

Neural networks and word embeddings have started to be incorporated into models for keyphrase extraction. For example, Wang et al. [56] investigated word embeddings to measure the relatedness between words in graph-based models. Marujo et al. [42] used word embeddings in the existing supervised MAUI system [43] to extract keywords from tweets. Mahata et al. [40] exploited word and phrase embeddings in an unsupervised topic-biased PageRank to extract keyphrases from research papers and showed improvements in performance over models that do not use embeddings. Bennani-Smires et al. [7] explored simple models for keyphrase extraction based on sentence embeddings. A Recurrent Neural Network (RNN) based approach was proposed by Zhang et al. [60] to identify keyphrases in Twitter data, using a joint-layer RNN to capture the semantic dependencies in the input sequence, but did not address the dependencies in the labels. Ray Chowdhury et al. [49] extended this joint-layer RNN to capture informal writing in tweets. Augenstein and Søgaard [5] used multi-task learning to classify keyphrase boundaries.

Inspired from work in machine translation, Meng et al. [44] focused on keyphrase generation (rather than keyphrase extraction) and addressed the task as a sequence to sequence learning problem, where the sequence of words in a document is used to generate a sequence of keyphrases. An Encoder-Decoder RNN, originally proposed by Cho et al. [15], was used to generate the keyphrase sequences. Several other works focused on keyphrase generation [13, 14, 57]. Unlike these works, we focus on keyphrase extraction and not keyphrase generation, which generates words that may or may not be present in the text. Specifically, we mine the content of documents to extract keyphrases that are present in their content, using CRF-based sequence labeling and the power of unsupervised word embeddings.

Sequence labeling models for keyphrase extraction have shown promising results in several studies [9, 22, 59]. For example, Gollapalli et al. [22] trained a CRF to extract keyphrases from scholarly documents, using features such as *tf-idf* and POS tags to predict a label for each token position in a document as being a keyphrase token (**KP**) or not (**Non-KP**). Recently, a sequence labeling framework has been explored on a variety of NLP tasks such as POS tagging, noun phrase chunking, named entity recognition, and

[1] The code and data are available upon request.

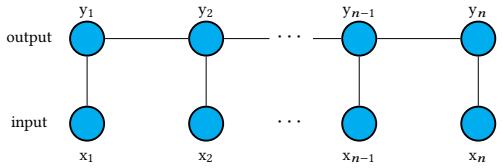

Figure 1: Layers in a CRF network.

keyphrase extraction [3, 26, 35, 39] that combines a Bi-LSTM network as the first layer to capture sequential text dependencies with a second CRF layer to capture label dependencies.

In our work, we explore word embeddings in CRF models as a simple and general approach for keyphrase extraction from scholarly documents, and contrast this with the more sophisticated model, Bi-LSTM-CRF. To our knowledge, in the context of keyphrase extraction from scholarly documents, we are the first to directly use word embeddings as features in CRF-based models and show improved results over previous models.

3 PROBLEM CHARACTERIZATION

CRF: We formulate keyphrase extraction as sequence labeling using Conditional Random Fields [31]. CRFs combine the advantages of graphical modeling and discriminative classification and have majors advantages over other graphical models, e.g., the ability to handle a large number of rich features and the ability to avoid the label bias problem (favoring the states with less outgoing transition) by using global normalization and accounting for the entire sequence at once. In CRFs, for an input sequence $\mathbf{x} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$, where each $\mathbf{x}_i$ represents the feature vector for the i^{th} word w_i in the sequence, the task is to predict a sequence of labels $\mathbf{y} = \{y_1, y_2, \cdots, y_n\}$, where y_i is assigned to w_i. In our CRF model, each feature vector $\mathbf{x}_i$ consists of individual or combinations of two types of features, document specific features and word embeddings, described below. Figure 1 shows a simple CRF network. The features that we use in our CRF models are described below.

Document Specific Features (DOC): We use six features for each word that are extracted from the target paper: a word's POS tag **(POS)** [27], term frequency-inverse document frequency **(tf-idf)** computed based on the target paper [20]; the position of the first occurrence of a word normalized by the length of the target paper (in the number of tokens) **(relative position)** [20, 27]; the distance of the first occurrence of a word from the beginning of a paper **(first position)** [12]; a binary feature indicating the presence of the word in the title **(is-in-title)** [28, 33]; and a binary feature indicating whether the word was capitalized **(is-capitalized)** [22]. The choice of these features was motivated by their good performance in previous models [12, 20, 27].

Word Embedding Features (EMB): Word embeddings are vector representations of words as dense vectors. Precisely, using word embeddings, words are expressed as dense vectors by projecting each word into a multidimensional vector space. The position of a word within the vector space or the embedding is learned from the text by considering the surrounding words from its local context, and hence, can capture the semantic relations from text. We use the components of multidimensional word embeddings of each word as its features.

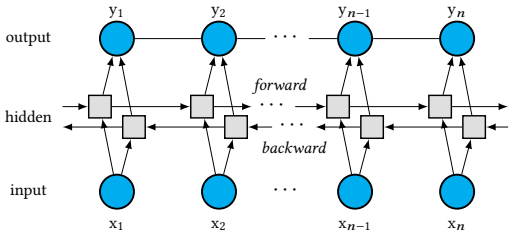

Figure 2: Layers in a Bi-LSTM-CRF network.

Bi-LSTM-CRF: In order to build a sequence labeling model that incorporates long distance information over a sequence of input as well as information on the output sequence, we consider the Bi-LSTM-CRF network as a more sophisticated and complex model. The network architecture is shown in Figure 2. As shown in the figure, the first layer of the model is a Bi-LSTM network with the purpose of capturing the semantics of the input text sequence. The output of the Bi-LSTM layer is passed to a CRF layer that produces a probability distribution over the tag sequence using the dependencies among labels of the entire sequence.

4 DATASETS

We used five datasets of research papers for evaluation. The first four datasets are widely used in keyphrase extraction and include SemEval-2010, Krapivin, Inspec, and NUS. The fifth dataset, called ACM, is a much larger dataset compared with the previous four datasets and is created from the collection of research papers published by ACM.[2] We used the title and abstract of each paper. The five datasets are described below.

(1) **SemEval** [29] contains 288 research papers from the ACM digital library along with author-assigned keyphrases. The dataset has a train and test split consisting of 188 and 100 papers, respectively.
(2) **Krapivin** [30] contains 2,304 ACM research papers with full text and author-assigned keyphrases. Similar to [44], since the dataset does not have a train-test split, we sampled 400 papers as the test set with the remaining papers being used as the training set.
(3) **Inspec** [27] contains abstracts of 2,000 research papers. It has a train-validation-test split of 1,000, 500 and 500 papers, respectively. In our experiments, we use the combination of train and validation sets to train different models.
(4) **NUS** [47] contains 211 research papers. This dataset does not have a train and test split and it is relatively small. Hence, consistent with [44], we performed five-fold cross-validation.
(5) **ACM:** Since none of the above datasets is very large, we constructed a new dataset of 30,000 papers published in ACM conferences. These papers were sampled from the 211,028 papers available in ACM Digital Library and published after 2010. In our experiments, we used 10,000 papers (at random) as the train set (**ACM-10k**) and the remaining 20,000 papers as the test set (**ACM-20k**). During the dataset construction, we ensured that there was no overlap between our ACM dataset and any of the four datasets above.

[2] https://dl.acm.org/

Dataset	Num. (#) Papers	Avg. # Keyphrases Per Paper	Number of Keyphrases of Different Lengths
SemEval	288	7.15	#unigrams: 440, #bigrams: 769, #trigrams: 383, # > trigrams:145
Krapivin	2,304	3.03	#unigrams: 1,476, #bigrams: 3,598, #trigrams: 1,210, # > trigrams:328
Inspec	2,000	7.65	#unigrams: 2,153, #bigrams: 7,068, #trigrams: 4,050, # > trigrams:1,955
NUS	211	2.71	#unigrams: 183, #bigrams: 258, #trigrams: 76, # > trigrams:16
ACM-30k	30,000	2.67	#unigrams: 30,658, #bigrams: 36,017, #trigrams: 10,916, # > trigrams:2,489

Table 1: Summary of the keyphrase extraction datasets used for model evaluation.

Dataset	Keyphrases
SemEval	congestion game, load-dependent failure, identical resource, nash equilibrium, nondecreasing cost function, potential function, failure probability, load-dependent failure, pure strategy nash equilibrium
Kapivin	bessel function, modified Bessel function, order zero and one, vectorized software
Inspec	evolving fuzzy rule-based models, identification, noniterative update, rule-base structure, incremental unsupervised learning, ranking, informative potential, fuzzy rules, complex processes, air-conditioning component modeling
NUS	Nearest Neighbour Search, TLAESA, Approximation Search
ACM-30k	Control abstractions, Data abstractions, Programming languages, Programming methodology

Table 2: Examples of gold standard keyphrases (present in the title + abstract) of a paper randomly selected from each dataset.

Table 1 shows a summary of all five datasets and contains the number of papers in each dataset, the average number of keyphrases per paper, and the number of n-gram keyphrases, for $n = 1, 2, 3$, and $n > 3$, for each collection. The gold-standard for each paper contains the author-assigned keyphrases present in a paper (its title and abstract). For finding gold keyphrases in a paper, we used the stemmed version of each. Table 2 shows examples of gold keyphrases (shown for one paper) from each dataset.

As described above and as can be seen from Tables 1 and 2, the datasets used for evaluation cover a wide variability in terms of dataset sizes, average number of keyphrases per paper, the length of keyphrases in the number of words.

5 EXPERIMENTAL DESIGN AND RESULTS

We performed three types of experiments. First, we evaluate the quality of word embeddings by themselves or combined with document specific features and contrast the learned CRFs with the CRFs that do not use word embeddings. Second, we contrast the CRF model that uses word embeddings and document features with a more sophiticated and complex model, Bi-LSTM-CRF. In this experiment, we also study the performance of CRF and Bi-LSTM-CRF when trained on the ACM-10k dataset and evaluated on the test set of each of the four datasets, SemEval, Krapivin, Inspec, and NUS. Third, we compare the CRF against several baselines and prior works, including supervised, sequence labeling and neural models. Finally, we show anecdotal evidence that demonstrates the quality of word embeddings in extracting appropriate keyphrases.

For our sequence labeling task, for model training, we convert a pair of the paper (title and abstract) and keyphrases pairs such that each sentence of a paper is a sequence of word tokens, each token has a positive label (**KP**) if it occurs in a keyphrase in the gold-standard, or a negative label (**Non-KP**), otherwise. Predicted keyphrases are obtained by combining all consecutive words predicted as **KP** (i.e., the longest sequence). We did not impose a constraint on the length of keyphrases since, as we can see from Table 1, particularly for Inspec and ACM-30k datasets, the number of keyphrases with length greater than three is very large.

Evaluation metrics. To evaluate the performance of the CRF models, we used the following metrics: precision, recall and F1-score for the positive class since the correct identification of positive examples (keyphrases) is more important. These metrics are widely used in previous works [12, 27, 45, 55]. To match the predicted keyphrases with gold-standard keyphrases, we do exact match between the stemmed version of each. For the NUS we perform 5-fold cross validation and for the other four datasets we perform a single train and test (on the train-test split provided by the authors of each dataset, or our train-test split for ACM). Because NUS is very small, cross-validation experiments are more appropriate on this dataset [44]. For the 5-fold cross validation experiments, the reported values are averaged across all (document level) folds.

5.1 Word Embeddings as Features in CRFs for Keyphrase Extraction

We contrast CRF models that use embedding features by themselves (EMB) and their combination with document specific features (EMB+DOC). We also contrast these models with the CRF models that use only document specific features (DOC) to understand the benefits of word embeddings in accurately predicting keyphrases. We also study the effect of embedding type (i.e., those trained on Google News and the ACM collection of research paper) and the embedding dimension.

We trained word embeddings of different dimension sizes (50, 100, 200, 300) on the ACM Digital Library collection of 211, 028 research papers using the Gensim implementation [50] of word2vec [46]. We kept words appearing in at least 5 documents for building the vocabulary. The full text has around 1.2B tokens and 879K unique tokens. We compared the ACM word embeddings (denoted "ACM") with the pre-trained word2vec embeddings of 300-dimensional vectors on Google News of about 100B words (denoted "GNews").

	DOC				EMB				DOC+EMB			
	Pr%	Re%	F1%	Emb.	Dim.	Pr%	Re%	F1%	Dim.	Pr%	Re%	F1%
SemEval	**36.18**	61.26	45.49	ACM	300	27.02	43.38	33.30	300	33.74	65.14	44.46
				GNews	300	28.41	50.28	36.30	300	34.38	**68.29**	**45.73**
Krapivin	33.99	54.25	41.79	ACM	300	24.39	37.52	29.56	300	33.62	63.76	44.02
				GNews	300	29.69	26.49	28.00	300	**38.95**	**64.64**	**48.61**
Inspec	46.86	74.48	57.52	ACM	300	46.37	64.50	53.95	300	49.36	83.34	62.00
				GNews	300	49.64	59.53	54.14	300	**51.73**	**84.91**	**64.29**
NUS	**32.44**	**53.46**	**40.38**	ACM	50	19.09	30.91	23.60	50	25.50	43.75	32.22
				GNews	300	18.82	30.08	23.16	300	26.52	49.18	34.46
ACM	37.41	55.00	44.53	ACM	300	27.15	29.28	28.18	300	**39.50**	**63.56**	**48.72**
				GNews	300	14.50	8.65	10.83	300	38.40	54.99	45.22

Table 3: CRF performance using word embeddings (EMB), document features (DOC) and DOC+EMB.

Table 3 shows the results of these comparisons using ACM and GNews with the best performing embedding dimension, for all five datasets. For **SemEval**, **Krapivin**, **Inspec**, and **ACM**, we used the train-test split for model training and evaluation, as described in Section 4, whereas for **NUS**, we used five-fold cross-validation.

As can be seen from Table 3, word embeddings alone (EMB) perform worse than the document specific features alone (DOC), for all datasets, in terms of most compared measures. For example, on the ACM dataset, EMB achieves an F1-score of 28.19% (using ACM embeddings) as compared with 44.53% achieved by DOC. When we add word embeddings to document features (DOC+EMB), we can see substantial improvements in performance over the individual features, DOC or EMB alone, for Krapivin, Inspec, and ACM, in terms of all compared measures, regardless of the embedding type used, ACM or GNews. For example, on ACM, DOC+EMB achieves an F1-score of 48.72% (using ACM embeddings), whereas DOC and EMB alone achieve an F1-score of 44.53% and 28.18%, respectively. However, on SemEval and NUS, DOC alone performs similarly or better than DOC+EMB, e.g., on SemEval, DOC achieves an F1-score of 45.49% as compared to 45.73% achieved by DOC+EMB using GNews embeddings, whereas on NUS, DOC achieves an F1-score of 40.38% as compared to 34.46% achieved by DOC+EMB using GNews embeddings. One potential explanation for this similar or drop in performance for SemEval and NUS is that both datasets have a relatively small size (see Table 1), with less than 200 papers for training in each dataset, that hinders to learn robust parameters for CRFs when the number of features is large as is the case with DOC+EMB, as compared with the number of DOC alone (only six features). These results show that word embeddings combined with document features help to improve the performance of CRF models when we have a sufficient amount of training data available.

From Table 3, we can also see that the embeddings trained on GNews perform similarly or better than those trained on ACM for DOC+EMB based CRFs on all datasets, except ACM. Also, for ACM embeddings, an embedding size of 300 generally works best among all compared sizes.

5.2 CRF vs. Bi-LSTM-CRF for Keyphrase Extaction

Next, we compare the CRF models with the more sophisticated Bi-LSTM-CRF models that are able to exploit the long-distance dependencies in the text. For the Bi-LSTM-CRF models, we used adam optimizer with learning rate 0.001, 300 cells for Bi-LSTM, and 30 epochs for model training.

Table 4 shows the results of this comparison for all five datasets. For model training and evaluation in this experiment, we used the train-test split of ACM, SemEval, Krapivin, and Inspec, and 5-fold cross-validation for NUS, with the embedding type and dimension that worked best on each dataset. The input of both CRFs and Bi-LSTM-CRFs is DOC+EMB. As can be seen from the table, CRFs consistently outperform the Bi-LSTM-CRFs on all five datasets. While the difference in F1-score is only 6.8% on ACM, it ranges between 20% and 40% on SemEval, Krapivin, and Inspec, which is attributed to the small size of these three datasets. Moreover, the difference in F1-score on NUS is low compared with the other datasets (only 1.8%). We believe that both CRF and Bi-LSTM-CRF are incapable of learning good model parameters due to the very small size of NUS.

To understand the impact of the training set size on the performance of the CRF and Bi-LSTM-CRF models on SemEval, Krapivin, Inspec, and NUS, and determine which model is a better for keyphrase extraction, we performed the following experiment: we trained both CRFs and Bi-LSTM-CRFs on the ACM-10k training set and evaluated their performance on the test splits of each SemEval, Krapivin, Inspec, and on the NUS dataset. Table 5 shows the results of this experiment using ACM word embeddings (300 dimensional).

From Table 5, we can make several observations. First, the performance of Bi-LSTM-CRF increases in terms of all compared measures on all four datasets when we train the model on a much larger dataset size, i.e., 10,000 research papers, compared with the relatively small sizes of the training set of each dataset, supporting our intuition that the Bi-LSTM-CRF model overfits in the experiments in Table 4. A similar trend is observed on NUS for the CRF model, suggesting that the small size of NUS hinders both models to learn robust parameters. Second, the difference in F1-score between the CRF abd Bi-LSTM-CRF is smaller when we train the models on ACM-10k. Third, we observe that the F1-score of CRF is higher than that of Bi-LSTM-CRF on three datasets, SemEval, Krapivin, and NUS, but is lower on Inspec. Interestingly, on Inspec, Bi-LSTM-CRF achieves a higher recall as compared with CRF, i.e., 59.4% vs. 31.1%. Inspec has a much higher average number of keyphrases per paper ($\approx$ 8 vs. 3) and contains a large number of keyphrases of length greater than 3 (1,955 n-grams with $n > 3$). Hence, the

ACM			Semeval			Krapivin			Inspec			NUS		
Pr%	Re%	F1%	Pr%	Re%	F1%	Pr%	Re%	F1%	Pr%	Re%	F1%	Pr%	Re%	F1%
CRF														
39.5	**63.5**	**48.7**	**34.3**	**68.2**	**45.7**	**38.9**	**64.6**	**48.6**	**51.7**	**84.9**	**64.2**	**26.5**	**49.1**	**34.4**
Bi-LSTM-CRF														
36.0	50.0	41.9	23.1	28.3	25.4	22.8	42.3	29.6	27.6	19.6	23.0	24.5	47.8	32.6

Table 4: Contrasting CRF with Bi-LSTM-CRF. The input to both models is DOC+EMB.

	Semeval			Krapivin			Inspec			NUS		
Train: ACM-10k	Pr%	Re%	F1%	Pr%	Re%	F1%	Pr%	Re%	F1%	Pr%	Re%	F1%
CRF	**43.9**	49.8	**46.7**	**37.8**	**58.0**	**45.7**	**42.5**	31.1	35.9	**38.2**	**67.3**	**48.7**
Bi-LSTM-CRF	32.2	**55.4**	40.7	27.8	54.1	36.7	36.0	**59.4**	**44.9**	26.4	56.4	35.9

Table 5: Performance of DOC+EMB on SemEval, Krapivin, Inspec and NUS while training on ACM-10k.

	SemEval			Krapivin			Inspec			NUS			ACM		
	Pr	Re	F1	Pr	Re	F1	Pr	Re	F1	Pr	Re	F1	Pr	Re	F1
CRF on DOC+EMB															
	34.4	**68.3**	**45.7**	**39.0**	**64.6**	**48.6**	51.7	**84.9**	**64.3**	26.5	49.2	34.5	**39.5**	**63.6**	**48.7**
Previous Models															
Hulth	28.7	10.7	15.6	21.2	8.4	12.1	27.5	70.5	39.6	20.2	12.4	15.4	20.0	14.2	16.6
KEA	25.6	10.0	14.4	22.2	10.1	13.9	26.6	5.1	8.5	25.3	24.2	24.7	22.6	17.0	19.4
Maui	27.4	38.1	31.9	20.0	56.0	29.5	24.2	34.2	28.4	20.3	54.7	29.6	22.0	63.2	32.7
CRF/PR	29.7	54.0	38.3	29.3	15.2	20.1	**56.9**	35.6	43.8	**33.2**	**61.4**	**43.1**	24.9	13.5	17.5
CopyRNN	24.7	39.5	30.4	17.6	54.2	26.6	29.2	40.9	34.1	26.6	42.1	32.6	17.8	42.2	25.0
Key2Vec	**38.6**	38.6	38.6	23.0	47.5	31.0	32.8	38.1	35.2	25.4	49.0	33.4	27.4	50.2	35.4

Table 6: The comparison of CRF that uses DOC+EMB with previous works. Performance is shown in %.

Bi-LSTM-CRF model, which exploits the long distance dependencies in the text, is able to accurately cover and identify these longer keyphrases. A similar result on recall is observed on SemEval (see Tables 1 and 2 for datasets characteristics and gold keyphrases).

Moreover, on Inspec, the performance of CRF when trained on ACM-10k (Table 5) is consistently smaller than that of CRF when trained on Inspec itself (the training portion). We believe this is due to the distribution of author-assigned keyphrases, which are different for Inspec and ACM-10k, and the size of Inspec training set is good enough to train a good CRF model.

5.3 Baseline Comparisons

Last, we compare the performance of CRF DOC+EMB using the best performing word embeddings and vector dimensions with several baseline approaches. Precisely, we compare the CRF DOC+EMB with six keyphrase extraction models: Hulth [27], KEA [20], Maui [43], the CRF model with posterior regularization (CRF/PR) from Gollapalli et al. [22], CopyRNN [44], and Key2Vec [40]. The choice of some of these models was motivated by their wide-spread usage as baselines in other related works [44], their good performance, and the integration of embeddings in existing systems as in Key2Vec. We used the implementations of Maui,[3] CRF/PR [22],[4] and CopyRNN,[5] and developed our implementation of Key2Vec.

Hulth uses POS tags, relative position, term frequency, and collection frequency as features. KEA uses tf-idf and relative position as features. Maui uses traditional and Wikipedia features, e.g., node degree, Wikipedia keyphraseness. The model proposed by Gollapalli et al. [22] uses CRF with posterior regularization, with three types of features: word, orthographic, and stopword features; parse-tree features; and title features, as well as their combinations incorporated with expert knowledge (i.e., predictions from other supervised and unsupervised methods). The CRF/PR model is the best performing model from Gollapalli et al. [22] and uses information solely from the document itself. CopyRNN uses an encoder-decoder Recurrent Neural Network with a copying mechanism as a generative model instead of extracting phrases only from the document text. We run CopyRNN for all five datasets with the parameter settings mentioned by the authors [44]. Key2Vec is a biased PageRank algorithm. We created a word graph by from nouns and adjectives and added an edge if two words appear within w words of each other in text. A *theme vector* for each paper was obtained by summing the embedding of the candidate words from its title. We used the GNews embeddings (300 dimension), which performed better than ACM in Key2Vec.

Table 6 shows the results of these comparisons for all five datasets, using the train-test split available for SemEval, Krapivin, Inspec, and ACM, and 5-fold cross-validation on NUS. As can be seen from the table, the DOC+EMB-based CRF that uses word embeddings in addition to document features substantially outperforms Hulth, KEA, Maui, CopyRNN, and Key2Vec, in terms of most compared measures, on four datasets, SemEval, Krapivin, Inspec, and ACM. For example, DOC+EMB based CRF achieves the highest F1-score of

[3] https://github.com/zelandiya/maui
[4] https://sites.google.com/site/sujathadas/home/pubslist
[5] https://github.com/memray/seq2seq-keyphrase

Incorporating **site-level knowledge** to extract structured data from **web forums**

Web forums have become an important data resource for many web applications, but extracting **structured data** from unstructured **web forum** pages is still a challenging task [...]. In this paper, we study the problem of **structured data** extraction from various **web forum** sites. Our target is to find a solution as general as possible to extract **structured data**, such as post title, post author, post time, and post content from any forum site. In contrast to most existing **information extraction** methods, which only leverage the knowledge inside an individual page, we incorporate both page-level and **site-level knowledge** and employ **Markov logic networks** (MLNs) [...]. The experimental results on 20 forums show a very encouraging **information extraction** performance, and demonstrate the ability of the proposed approach on various forums. [...]

Incorporating site-level **knowledge** to extract structured data from **web forums**

Web forums have become an important **data** resource for many **web** applications, but **extracting structured data** from unstructured **web forum** pages is still a challenging task [...]. In this paper, we study the problem of structured **data extraction** from various **web forum sites**. Our target is to find a solution as general as possible to extract structured **data**, such as post title, post author, post time, and post content from any **forum site**. In contrast to most existing **information extraction** methods, which only leverage the knowledge inside an individual page, we incorporate both page-level and site-level **knowledge** and employ **Markov logic networks** (MLNs) [...]. The experimental results on 20 forums show a very encouraging **information extraction** performance, and demonstrate the ability of the proposed approach on various forums. [...]

Human-input keyphrases: *Web forums, Structured data, Information extraction, Site level knowledge, Markov logic networks*

Predicted keyphrases: ***Web forums***, ***Information extraction***, ***Markov logic networks***, *Extracting structured data, Data, Knowledge, Data extraction, Web, Forum site, Web forum sites*

Figure 3: The title, abstract, human-input keyphrases and predicted keyphrases of an ACM paper. The phrases marked with cyan in the title and abstract shown on the top of the figure are gold keyphrases, whereas the words and phrases marked with dark blue in the title and abstract shown on the bottom of the figure are predicted keywords/keyphrases.

64.3% on Inspec. Interestingly, on NUS, CRF/PR achieves the highest F1-score of 43.1% among all models. Moreover, CRF/PR achieves the highest precision on Inspec dataset, and the highest value among all measures on NUS. Key2Vec achieves the highest precision of 38.6% on SemEval. It is worth noting that the DOC+EMB based CRF models yield improvements in performance over more complex deep learning model, CopyRNN, on all five datasets.

5.4 Anecdotal Evidence

To see the quality of predicted phrases by the best EMB+DOC based CRF, we randomly selected a paper from our ACM dataset and evaluated the CRF model on it. Note that this selected paper belongs to the test portion of the dataset. We manually inspected the CRF predictions and contrasted them with the author-annotated (gold) keyphrases. The title, abstract, human annotated keyphrases and predicted keyphrases for this paper are shown in Figure 3. Specifically, the cyan bold phrases shown in the text on the top of the figure represent author-assigned keyphrases, whereas the dark blue bold phrases shown in the text on the bottom of the figure represent the positively predicted phrases using the CRF trained on DOC+EMB features. It can be seen from the figure that the predicted keyphrases cover three out of five author assigned keyphrases, and for the remaining two author assigned keyphrases the CRF model predicted one super-string and one substring.

We can also observe that the model was not able to predict "structured data" nor "site level knowledge" even though both these gold keyphrases appear in the title of the document. However, for the two gold keyphrases missed by the model, the model predicted "extracting structured data", "data", and "knowledge" as keyphrases which are a super-string or substring of them. Moreover, predicted keyphrases "web", "forum site", and "web forum sites" are related to one of the author-assigned keyphrase "web forums," which was correctly predicted by the DOC+EMB CRF model. As there exist a semantic relation between "data" and "information," DOC+EMB is able to predict "data extraction," which is similar with the author-assigned keyphrase "information extraction".

6 CONCLUSIONS AND FUTURE DIRECTIONS

In this paper, we explored keyphrase extraction as sequence labeling using CRFs and studied the benefits of using word embeddings in conjunction with document specific features in order to capture the semantics of words in context. Our results showed interesting performance variability, according to training set sizes, the number of keyphrases per document, and their length in the number of words, but clearly highlighted the benefits of using word embeddings as features in CRFs along with document specific features. Our results also showed improvements in performance over complex models including more sophisticated deep learning models.

In the future, it would be interesting to integrate posterior regularization in the word embeddings based CRF models. It would also be interesting to explore keyphrase extraction from research papers from other fields in Computer Science such as Computational Linguistics, as well as other scientific domains, such as Biology, Social Science, Political Science, and Material Sciences. Moreover, since these scientific domains do not generally have author-annotated keyphrases, developments of domain adaptation and transfer learning techniques should also be investigated.

ACKNOWLEDGMENTS

This research is supported by NSF awards 1802358 and 1823292. The ACM Digital Library dataset is provided for non-commercial

research purposes, courtesy of the Association for Computing Machinery, Inc. [2018].

REFERENCES

[1] Amjad Abu-Jbara and Dragomir Radev. 2011. Coherent Citation-based Summarization of Scientific Papers. In *ACL: HLT*. 500–509.

[2] Eytan Adar and Srayan Datta. 2015. Building a Scientific Concept Hierarchy Database (SCHBase). In *ACL*. 606–615.

[3] Rabah Alzaidy, Cornelia Caragea, and C Lee Giles. 2019. Bi-LSTM-CRF Sequence Labeling for Keyphrase Extraction from Scholarly Documents. In *WWW*. ACM, 2551–2557.

[4] Isabelle Augenstein, Mrinal Das, Sebastian Riedel, Lakshmi Vikraman, and Andrew McCallum. 2017. SemEval 2017 Task 10: ScienceIE - Extracting Keyphrases and Relations from Scientific Publications. In *SemEval@ACL*. 546–555.

[5] Isabelle Augenstein and Anders Søgaard. 2017. Multi-Task Learning of Keyphrase Boundary Classification. In *ACL*, Vol. 2. 341–346.

[6] Ken Barker and Nadia Cornacchia. 2000. Using noun se heads to extract document keyphrases. (2000), 40–52.

[7] Kamil Bennani-Smires, Claudiu Musat, Andreea Hossmann, Michael Baeriswyl, and Martin Jaggi. 2018. Simple Unsupervised Keyphrase Extraction using Sentence Embeddings. In *CoNLL*. 221–229.

[8] Gábor Berend. 2011. Opinion Expression Mining by Exploiting Keyphrase Extraction. In *Proceedings of 5th International Joint Conference on Natural Language Processing*. 1162–1170.

[9] Pinaki Bhaskar, Kishorjit Nongmeikapam, and Sivaji Bandyopadhyay. 2012. Keyphrase Extraction in Scientific Articles: A Supervised Approach. In *COLING*. Mumbai, India, 17–24.

[10] Ido Blank, Lior Rokach, and Guy Shani. 2013. Leveraging the Citation Graph to Recommend Keywords. In *RecSys*. 359–362.

[11] Florin Bulgarov and Cornelia Caragea. 2015. A Comparison of Supervised Keyphrase Extraction Models. In *WWW*. 13–14.

[12] Cornelia Caragea, Florin Adrian Bulgarov, Andreea Godea, and Sujatha Das Gollapalli. 2014. Citation-enhanced keyphrase extraction from research papers: A supervised approach. In *EMNLP*. 1435–1446.

[13] Jun Chen, Xiaoming Zhang, Yu Wu, Zhao Yan, and Zhoujun Li. 2018. Keyphrase Generation with Correlation Constraints. In *EMNLP*. 4057–4066.

[14] Wang Chen, Yifan Gao, Jiani Zhang, Irwin King, and Michael R. Lyu. 2018. Title-Guided Encoding for Keyphrase Generation. *CoRR* abs/1808.08575 (2018).

[15] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv:1406.1078* (2014).

[16] Soheil Danesh, Tamara Sumner, and James H Martin. 2015. SGRank: Combining Statistical and Graphical Methods to Improve the State of the Art in Unsupervised Keyphrase Extraction. *Lexical and Computational Semantics* (2015), 117.

[17] Samhaa R El-Beltagy and Ahmed Rafea. 2010. Kp-miner: Participation in semeval-2. In *SemEval*. 190–193.

[18] Wei Fan, Huan Liu, Suge Wang, Yuxiang Zhang, and Yaocheng Chang. 2019. Extracting Keyphrases from Research Papers Using Word Embeddings. In *Advances in Knowledge Discovery and Data Mining*. 54–67.

[19] Corina Florescu and Cornelia Caragea. 2017. PositionRank: An Unsupervised Approach to Keyphrase Extraction from Scholarly Documents. In *ACL*. Vancouver, Canada, 1105–1115.

[20] Eibe Frank, Gordon W Paynter, Ian H Witten, Carl Gutwin, and Craig G Nevill-Manning. 1999. Domain-specific keyphrase extraction. In *IJCAI*. 668–673.

[21] Sujatha Das Gollapalli and Cornelia Caragea. 2014. Extracting keyphrases from research papers using citation networks. In *AAAI*. 1629–1635.

[22] Sujatha Das Gollapalli, Xiao-Li Li, and Peng Yang. 2017. Incorporating Expert Knowledge into Keyphrase Extraction.. In *AAAI*. 3180–3187.

[23] Maria Grineva, Maxim Grinev, and Dmitry Lizorkin. 2009. Extracting key terms from noisy and multitheme documents. In *WWW*. 661–670.

[24] Kazi Saidul Hasan and Vincent Ng. 2010. Conundrums in unsupervised keyphrase extraction: making sense of the state-of-the-art. In *COLING*. 365–373.

[25] Kazi Saidul Hasan and Vincent Ng. 2014. Automatic Keyphrase Extraction: A Survey of the State of the Art. In *ACL*. 1262–1273.

[26] Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *arXiv preprint arXiv:1508.01991* (2015).

[27] Anette Hulth. 2003. Improved automatic keyword extraction given more linguistic knowledge. In *EMNLP*. 216–223.

[28] Xin Jiang, Yunhua Hu, and Hang Li. 2009. A ranking approach to keyphrase extraction. In *SIGIR*. 756–757.

[29] Su Nam Kim, Olena Medelyan, Min-Yen Kan, and Timothy Baldwin. 2010. SemEval-2010 Task 5: Automatic Keyphrase Extraction from Scientific Articles. In *SemEval*. 21–26.

[30] Mikalai Krapivin, Aliaksandr Autaeu, and Maurizio Marchese. 2009. *Large dataset for keyphrases extraction.* Technical Report. University of Trento.

[31] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *ICML*. 282–289.

[32] Shibamouli Lahiri, Sagnik Ray Choudhury, and Cornelia Caragea. 2014. Keyword and Keyphrase Extraction Using Centrality Measures on Collocation Networks. *CoRR* abs/1401.6571 (2014). http://arxiv.org/abs/1401.6571

[33] Marina Litvak and Mark Last. 2008. Graph-based keyword extraction for single-document summarization. In *Proceedings of the workshop on Multi-source Multilingual Information Extraction and Summarization*. 17–24.

[34] Feifan Liu, Deana Pennell, Fei Liu, and Yang Liu. 2009. Unsupervised approaches for automatic keyword extraction using meeting transcripts. In *ACL*. 620–628.

[35] Liyuan Liu, Jingbo Shang, Frank Xu, Xiang Ren, Huan Gui, Jian Peng, and Jiawei Han. 2018. Empower Sequence Labeling with Task-Aware Neural Language Model. In *AAAI*.

[36] Zhiyuan Liu, Wenyi Huang, Yabin Zheng, and Maosong Sun. 2010. Automatic keyphrase extraction via topic decomposition. In *EMNLP*. 366–376.

[37] Zhiyuan Liu, Peng Li, Yabin Zheng, and Maosong Sun. 2009. Clustering to find exemplar terms for keyphrase extraction. In *EMNLP*. 257–266.

[38] Patrice Lopez and Laurent Romary. 2010. HUMB: Automatic key term extraction from scientific articles in GROBID. In *SemEval*. 248–251.

[39] Xuezhe Ma and Eduard Hovy. 2016. End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF. In *ACL*. Berlin, Germany, 1064–1074. http://www.aclweb.org/anthology/P16-1101

[40] Debanjan Mahata, John Kuriakose, Rajiv Ratn Shah, and Roger Zimmermann. 2018. Key2Vec: Automatic Ranked Keyphrase Extraction from Scientific Articles using Phrase Embeddings. In *NAACL*. 634–639.

[41] Juan Martinez-Romo, Lourdes Araujo, and Andres Duque Fernandez. 2016. SemGraph: Extracting keyphrases following a novel semantic graph-based approach. *JASIST* 67, 1 (2016), 71–82.

[42] Luis Marujo, Wang Ling, Isabel Trancoso, Chris Dyer, Alan W Black, Anatole Gershman, David Martins de Matos, João Neto, and Jaime Carbonell. 2015. Automatic Keyword Extraction on Twitter. In *ACL and IJCNLP*.

[43] Olena Medelyan, Eibe Frank, and Ian H Witten. 2009. Human-competitive tagging using automatic keyphrase extraction. In *EMNLP*. 1318–1327.

[44] Rui Meng, Sanqiang Zhao, Shuguang Han, Daqing He, Peter Brusilovsky, and Yu Chi. 2017. Deep Keyphrase Generation. In *ACL*. 582–592.

[45] Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing Order into Text. In *EMNLP*. 404–411.

[46] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS*. 3111–3119.

[47] Thuy Dung Nguyen and Min-Yen Kan. 2007. Keyphrase extraction in scientific publications. In *Asian Digital Libraries*. 317–326.

[48] Vahed Qazvinian, Dragomir R. Radev, and Arzucan Özgür. 2010. Citation Summarization Through Keyphrase Extraction. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING '10)*. 895–903.

[49] Jishnu Ray Chowdhury, Cornelia Caragea, and Doina Caragea. 2019. Keyphrase Extraction from Disaster-related Tweets. In *The World Wide Web Conference (WWW '19)*. 1555–1566.

[50] Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *LREC*. 45–50.

[51] Lucas Sterckx, Cornelia Caragea, Thomas Demeester, and Chris Develder. 2016. Supervised Keyphrase Extraction as Positive Unlabeled Learning. In *EMNLP*. 1924–1929.

[52] Nedelina Teneva and Weiwei Cheng. 2017. Salience Rank: Efficient Keyphrase Extraction with Topic Modeling. In *ACL*, Vol. 2. 530–535.

[53] Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word Representations: A Simple and General Method for Semi-supervised Learning. In *ACL*. 384–394.

[54] Peter D Turney. 2003. Coherent keyphrase extraction via web mining. *arXiv preprint cs/0308033* (2003).

[55] Xiaojun Wan and Jianguo Xiao. 2008. Single Document Keyphrase Extraction Using Neighborhood Knowledge.. In *AAAI*. 855–860.

[56] Rui Wang, Wei Liu, and Chris McDonald. 2014. Corpus-independent Generic Keyphrase Extraction Using Word Embedding Vectors. In *Software Engineering Research Conference*. 39.

[57] Hai Ye and Lu Wang. 2018. Semi-Supervised Learning for Neural Keyphrase Generation. In *EMNLP*. 4142–4153.

[58] Wen-tau Yih, Joshua Goodman, and Vitor R. Carvalho. 2006. Finding Advertising Keywords on Web Pages. In *WWW'06*. 213–222.

[59] Chengzhi Zhang, Huilin Wang, Yao Liu, Dan Wu, Yi Liao, and Bo Wang. 2008. Automatic Keyword Extraction from Documents Using Conditional Random Fields. *JCIS* 4(3) (2008), 1169–1180.

[60] Qi Zhang, Yang Wang, Yeyun Gong, and Xuanjing Huang. 2016. Keyphrase extraction using deep recurrent neural networks on Twitter. In *EMNLP*. 836–845.

[61] Yongzheng Zhang, Evangelos Milios, and Nur Zincir-Heywood. 2007. A comparative study on key phrase extraction methods in automatic web site summarization. *JDIM* 5, 5 (2007), 323.