SurfKE: A Graph-Based Feature Learning Framework for Keyphrase Extraction

Corina Florescu

Dissertation Prepared for the Degree of
DOCTOR OF PHILOSOPHY
University of North Texas

May 24th, 2019

Major Professor: Dr. Wei Jin
Committee Members: Dr. Eduardo Blanco
Dr. Rodney Nielsen
Dr. Song Fu
Dr. Yan Huang

Outline I

Outline

Problem Definition

Text Selected from "Markov Chain" Wikipedia Page

A **Markov chain** is a stochastic model describing a sequence of possible events in which the probability of each event depends only on the state attained in the previous event. In probability theory and related fields, a **Markov process**, named after the Russian mathematician Andrey Markov, is a **stochastic process** that satisfies the Markov property. A **Markov chain** is a type of **Markov process** that has either discrete state space or discrete index set, [...]. Random walks on integers and the gambler's ruin problem are examples of Markov processes[...]. **Markov chains** have many applications as statistical models of real-world processes, [...]. The algorithm known as **PageRank**, which was originally proposed for the Internet search engine Google, is based on a **Markov process**.

- **Potential Keyphrases:**
  Markov chain, Markov process, stochastic process

Outline

Background and Motivation

- Keyphrases are very useful in many natural language processing, information retrieval and data mining tasks (Hulth and Megyesi, 2006; Berend, 2011)
- Their importance is emphasized by NLP and ML packages such as:

```
>>> from gensim.summarization import keywords
>>> text = '''Challenges in natural language processing frequently involve
... speech recognition, natural language understanding, natural language
... generation (frequently from formal, machine-readable logical forms),
... connecting language and machine perception, dialog systems, or some
... combination thereof.'''
>>> keywords(text).split('\n')
[u'natural language', u'machine', u'frequently']
```

Background and Motivation

- The effectiveness and utility of these applications depend on both the availability and quality of keyphrases.
- The performance of keyphrase extraction systems is much lower than that of other NLP tasks. (*e.g. part-of-speech tagging (over 90%) while keyphrase extraction (around 35%)*)
- Additional research on keyphrase extraction is needed

Outline

Contributions

Expected contributions and benefits:

- Graph-based Representations for Keyphrase Extraction
- Learning to Rank Keyphrases based on Multi-Scale Ratings
- Data Collection
- `https://github.com/boudinfl/pke`

pke - python keyphrase extraction

pke is an **open source** python-based **keyphrase extraction** toolkit. It provides an end-to-end keyphrase extraction pipeline in which each component can be easily modified or extended to develop new models. pke also allows for easy benchmarking of state-of-the-art keyphrase extraction models, and ships with supervised models trained on the SemEval-2010 dataset.

build passing

Implemented models

pke currently implements the following keyphrase extraction models:

- Unsupervised models
  - Statistical models
    - TfIdf [documentation]
    - KPMiner [documentation, article by (El-Beltagy and Rafea, 2010)]
    - YAKE [documentation, article by (Campos et al., 2018)]
  - Graph-based models
    - TextRank [documentation, article by (Mihalcea and Tarau, 2004)]
    - SingleRank [documentation, article by (Wan and Xiao, 2008)]
    - TopicRank [documentation, article by (Bougouin et al., 2013)]
    - TopicalPageRank [documentation, article by (Sterckx et al., 2015)]
    - PositionRank [documentation, article by (Florescu and Caragea, 2017)]
    - MultipartiteRank [documentation, article by (Boudin, 2018)]
- Supervised models
  - Feature-based models
    - Kea [documentation, article by (Witten et al., 2005)]
    - WINGNUS [documentation, article by (Nguyen and Luong, 2010)]

Outline

Supervised Approaches

Many approaches to keyphrase extraction have been proposed in the literature along two lines of research: *supervised* and *unsupervised*.

- Supervised approaches
  - Formulated as a binary classification problem
  - Different features and classification algorithm give rise to different models (**Frank et al., 1999**; Hulth, 2003; **Medelyan et al., 2009**; Lopez and Romary, 2010; Caragea et al., 2014; Gollapalli et al., 2017).

Common Features
Term Frequency
TF-IDF
Position of the first occurrence
Spread
Keyphraseness
Phrase length
Wikipedia-based keyphraseness
Inverse Wikipedia linkage
Part-of-Speech Tag

Outline

Unsupervised Approaches

- Unsupervised approaches
  - formulated as a ranking problem where phrases are ranked based on various measures *e.g., tf-idf, graph-based ranking methods*
  - Heuristics-based approaches (Zhang et al., 2007; **El-Beltagy and Rafea, 2010**)
  - Graph-based approaches (Mihalcea and Tarau, 2004; Wan and Xiao, 2008; Liu et al., 2009; **Bougouin et al., 2013**; Gollapalli and Caragea, 2014; Wang et al., 2014; **Florescu and Caragea, 2017**)

Outline

Graph Representation Learning

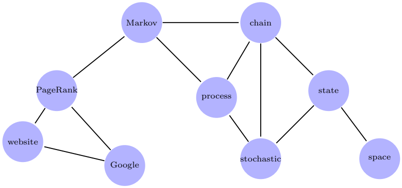
- What is graph representation learning?

Graph Representation Learning

- What is graph representation learning?
- Why using graph representation learning for keyphrase extraction?

Graph Representation Learning

- What is graph representation learning?
- Why using graph representation learning for keyphrase extraction?



Outline

SurfKE

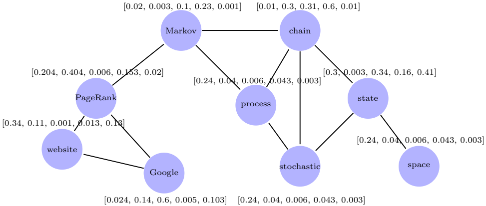
- SurfKE involves four main steps:
  - the graph construction at word level;
  - the process of learning continuous feature representations;
  - the formation of candidate phrases;
  - the process of training the model.

Word Graph Construction $G = (V, E)$

- Each unique word that passes certain part-of-speech tags corresponds to a vertex $v_i \in V$.
- Two vertices v_i and v_j are linked by an edge $(v_i, v_j) \in E$ if the two words co-occur within a fixed window of w contiguous tokens.
- For each node $v_i \in V$, we compute a set of attributes (features) as follows:
  - Tf-idf.
  - First Position
  - Spread
  - PageRank
  - Word-Document Similarity

$$sim_{word-doc} = \frac{1}{|doc|} \sum_{word \in doc} tfidf(word) * word2vec(word)$$

A Toy Example of a Word Graph

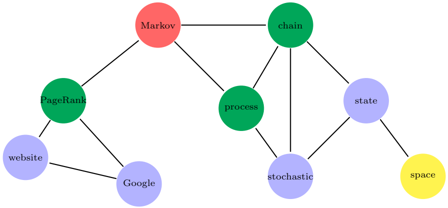


Feature Learning Framework

Let $G = (V, E)$ be a word graph built as explained before.

- **Goal:** learn a function that associates with each node $v_i \in V$ a vector representation that preserves some graph properties while integrating the existing node features
- We learn node embeddings based on the information available in a node's neighborhood (feature of nodes, structure of the graph).
- We define the local neighborhood of a node as the set of nodes up to k hops away from a given node where k is a parameter to be tuned.

Feature Learning Framework



- Train a set of functions (neural networks) that learn to aggregate feature information from a node's local neighborhood
- Nodes in the graph have different embeddings at each layer and initially (at layer zero) they are set to the node attributes

Feature Learning Framework

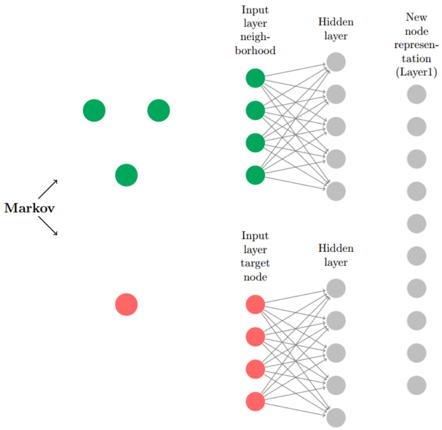
Let z_v^k be the node representation at step k and $N(v)$ the neighborhood of node v. Then, the embedding of a node v at step k is defined as:

$$z_v^k = \phi([W_K AGGREGATE(\{h_u^{k-1}, \forall u \in N(v)\}), B_K z_v^{k-1}])$$

- $AGGREGATE(\{h_u^{k-1}, \forall u \in N(v)\}$ is achieved by taking the element-wise mean of the neighbors
- W_K and B_K are the neural networks' weights and ϕ is a non-linear function (e.g., ReLU).

Feature Learning Framework

$$z_v^k = \phi([W_K \sum_{u \in N(v)} \frac{z_u^{k-1}}{|N(v)|}, B_K z_v^{k-1}])$$



Feature Learning Framework

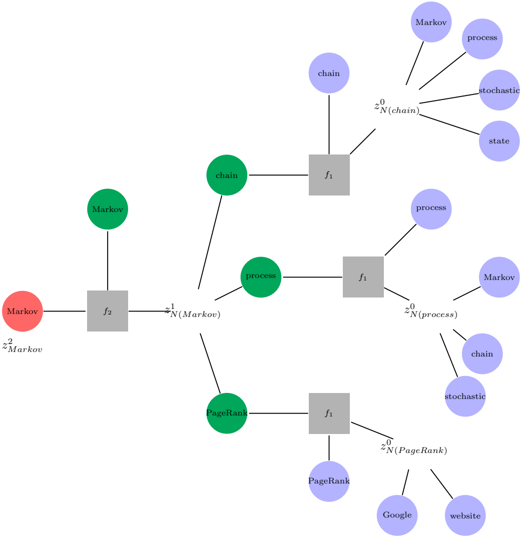
- We learn the weight matrices, W_k, $K = 1, k$ via stochastic gradient descent by applying a loss function to the output representations (z_v).

- The most common way to measure similarity between nodes in the graph is based on random walks.

$$J(z_v) = -log(\sigma(z_u^T z_u)) - QE_{v_n \sim P_n(v)} log(\sigma(-z_u^T z_{v_n}))$$

  - σ is the sigmoid function
  - P_n is a negative sampling distribution
  - Q defines the number of negative samples.
  - $P_n = \frac{f(v_i)^\beta}{\sum_j^n (f(v_i)^\beta)}$

Feature Learning Framework



Forming Candidate Phrases

- Candidate words that have contiguous positions in a document are concatenated into phrases.
- We extract noun phrases with pattern (adjective)*(noun)+ of length up to five tokens. We regard these noun phrases as candidate keyphrases.
- The feature vector for a multi-word phrase (e.g., *keyphrase extraction*) is obtained by taking the component-wise mean of the vectors of words constituting the phrase.

Data

- DUC - news articles
- Inspec - abstracts of research papers
  relatively popular dataset for automatic keyphrase extraction
- Medline - medical research papers from the MEDLINE/PubMed database[1]

Data	**#Docs**	**AvgKp**	**$\|$kp$\|$=1**	**$\|$kp$\|$=2**	**$\|$kp$\|$=3**	**$\|$kp$\| \geq$ 4**
DUC	308	8.08	17.31%	61.25%	17.71%	3.73%
Inspec	2000	9.63	42.72%	40.14%	11.93%	5.1%
Medline	500	3.00	42.73%	39.46%	13.00%	4.8%

Table: A summary of our datasets.

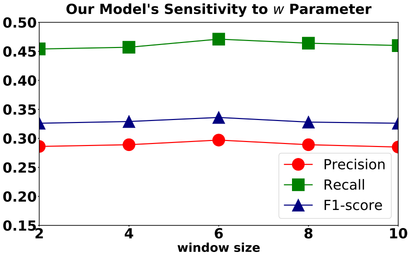
[1] https://www.nlm.nih.gov/databases/download/pubmed_medline.html

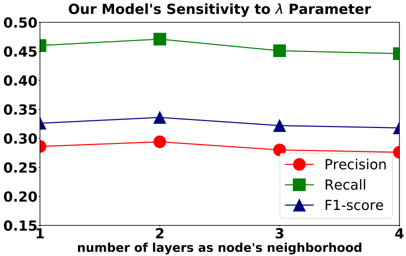
Outline

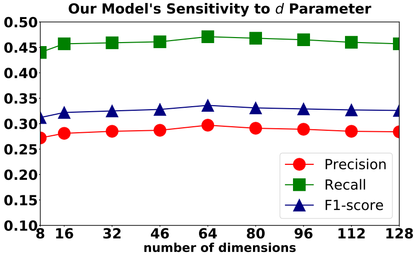
Experimental Design

- We investigate the performance of our model on DUC and Inspec separately, and on the synthesized data (DUC + Inspec + Medline)
- We randomly split the data into 75% training and 25% testing at the document level
- The model parameters were estimated on the training set using a 10-fold cross-validation setting.
- We measure the performance of SurfKE using Precision, Recall and F1-score.

Parameter Sensitivity







Comparison with Baselines

- Supervised Approaches
  - KEA (Frank et al., 2001)
  - Maui (Medelyan et al., 2009)
  - Input Features
- Unsupervised Approaches
  - KpMiner (El-Beltagy et al., 2010)
  - TopicRank (Bougouin et al., 2013)
  - PositionRank (Florescu and Caragea, 2017)

Comparison with Baselines

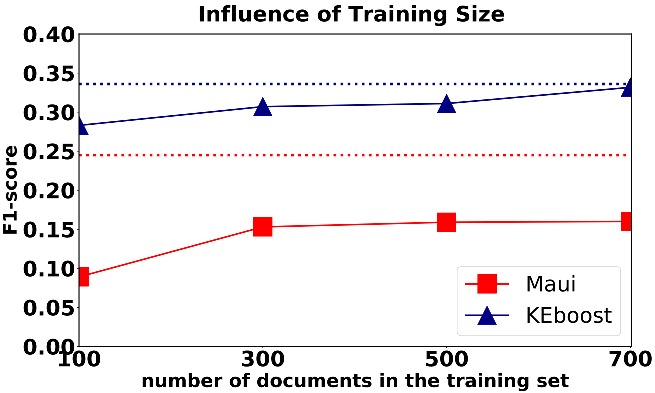
Approach	Precision	Recall	F1-score
KEA	0.146	0.247	0.168
Maui	0.207	0.362	0.245
Input Features	0.267	0.450	0.310
SurfKE	**0.297**	**0.471**	**0.336**

Table: The performance of SurfKE in comparison with the supervised models on the heterogeneous data.

Approach	Precision	Recall	F1-score
KpMiner	0.182	0.335	0.219
TopicRank	0.230	0.356	0.261
PositionRank	0.231	0.389	0.270
SurfKE	**0.297**	**0.471**	**0.336**

Table: The performance of SurfKE in comparison with the unsupervised models on the heterogeneous collection.

The Training Size Impact on the Performance of SurfKE



Error Analysis

- A system incorrectly predicts several candidates as keyphrases because they contain a word that represents the main topic of the target document; e.g., 'GPRS" and "GPRS phones".
- A model predicts a candidate as a keyphrase; e.g., sprinter Ben Johnson and Ben Johnson.
- A model correctly predicts a keyphrase but the gold standard is a alternative form of that concept; e.g.,"link prediction" "link prediction model".

Outline

SurfKing:A Learning-to-Rank Framework using Multi-Scale Ratings

- Formulating KE as a binary classification problem has a couple of drawbacks:
  - a classifier identifies keyphrases independently of the other candidates in the document and consequently, it cannot determine which candidates are more relevant to the topic of a given document.
  - the goal of keyphrase extraction is to identify the most representative phrases for a document rather than classifying phrases in an absolute sense.
- Jiang et al., 2009 proposed a learning to rank approach to address this issue

SurfKing:A Learning-to-Rank Framework using Multi-Scale Ratings

We posit that not all keyphrases of a document have the same level of appropriateness in relation to the topic of the document.

> Canadian **Ben Johnson** left the **Olympics** today "in a complete state of shock," accused of cheating with drugs in the world's fastest **100-meter dash** and stripped of his **gold medal**. The prize went to American **Carl Lewis**. Many athletes accepted the accusation that Johnson used a muscle-building but dangerous and illegal anabolic steroid called **stanozolol** as confirmation of what they said they know has been going on in track and field. Two tests of Johnson's urine sample proved positive and his denials of **drug use** were rejected today. "This is a blow for the Olympic Games and the Olympic movement," said International Olympic Committee President Juan Antonio Samaranch.

Figure: An anecdotal example.

A finer measurement of a phrase's appropriateness in the extraction process may yield a more accurate list of keyphrases.

Learning-to-Rank

- Learning to Rank (LTR) is a class of techniques that apply supervised machine learning (ML) to solve ranking problems.
- Regression $\neq$ Learning-to-Rank (LTR) because LTR does not need to predict the absolute value of an instance.
- Classification $\neq$ Learning-to-Rank (LTR) because LTR does not need to predict the absolute class of an instance.
- Learning to Rank - The relative ranking of items is all that is important.
- LTR solves a ranking problem on a list of items. The aim of LTR is to come up with optimal ordering of those items.

Learning to Rank - Pairwise Approach

- Ranking is transformed into a pairwise classification or regression problem.
- Let $X \subset \mathbb{R}^n$ be the set of input feature vectors, $Y = \{y_1, y_2, ..., y_m\}$ the target ranks and f a linear ranking function such as $f = w^T x$.
  $f(x_i) > f(x_j) \iff w^T x_i > w^T x_j \iff w^T(x_i - x_j) > 0 \iff y_i > y_j$
- Then $x_i - x_j$ can be considered as a positive example and assign $+1$ if $y_i > y_j$ or as a negative example (-1) otherwise.

RankNet

- There are several algorithms that can be used to learn the ranking function with SVM^{Rank} being the most popular.
- We chose to explore RankNet for our task, since it is usually implemented using a neural network which is more flexible and can approximate more complex bounded continuous functions.
- RankNet maps each input feature vector $x \in R^n$ to a number $f(x)$. For instance, $s_i = f(x_i)$ and $s_j = f(x_j)$.
- Then, the two outputs of the model are mapped to a learned probability:

$$P_{ij} = \frac{e^{(s_i - s_j)}}{1 + e^{(s_i - s_j)}}$$

- Cross-entropy cost function:

$$C = \bar{P}_{ij} \log(P_{ij}) - (1 - \bar{P}_{ij}) \log(P_{ij}))$$

Outline

Learning-to-Rank Keyphrases

- $D = \{d_1, d_2, ..., d_p\}$ a collection of documents where $p = |D|$
- $C_d = \{c_{1,d}, c_{2,d}, ..., c_{n,d}\}$ a set of candidate keyphrases for d
- Each $c_{i,d} \in C_d$ is associated with a rank that reflects the degree to which $c_{i,d}$ is relevant to the topic of document d
- Let $L = \{l_1, l_2, ..., l_m\}$ be the set of all relevance degrees (levels) such that there is a total order among them.
- $x_{i,d}$ is the feature vector of candidate $c_{i,d}$ and $l_{i,d}$ is the target value of candidate $c_{i,d}$.
- Two examples are considered for a pairwise preference constraint if (1) they appear within the same document; (2) they are assigned different relevance ranks.

Learning-to-Rank Keyphrases

Data	Features			Relevance
x_{1,d_1}	0.343	0.237	0.018	3
x_{2,d_1}	0.571	0.113	0.873	1
x_{3,d_1}	0.456	0.092	0.034	2
x_{4,d_1}	0.620	0.022	0.378	2
x_{1,d_2}	0.456	0.092	0.034	2
x_{2,d_2}	0.620	0.022	0.378	1

(a)

Data	Features			Relevance
$x_{1,d_1} - x_{2,d_1}$	-0.228	0.124	-0.855	2 (3-1)
$x_{1,d_1} - x_{3,d_1}$	-0.113	0.145	-0.016	1 (3-2)
...	...	...	...	...
...	...	...	...	...
$x_{4,d_1} - x_{2,d_1}$	0.049	-0.091	-0.495	1 (2-1)
$x_{1,d_2} - x_{2,d_2}$	0.049	-0.091	-0.495	-1 (1-2)

(b)

Data

- We annotated with phrase importance the two benchmark datasets DUC and Inspec (300 documents)
- We annotated the datasets with the help of two graduate students which were carefully instructed to assign the scores.
- For each document, we extracted all phrases with pattern (adjective)*(noun)+ then assigned to each phrase an integer score:
  - **0** (zero) or *not related* - the phrase does not represent the topics of that document;
  - **1** or *weakly related*- the phrase is weakly related to the document;
  - **2** or *related* - the phrase links to at least one of the topics of the document;
  - **3** or *strong related* - the phrase represents the main topic of the document;

Data

In probability theory^{1} and related fields^{0}, a Markov process^{3}, named after the Russian mathematician Andrey Markov^{1}, is a stochastic process^{2} that satisfies the Markov property^{2} [...]. A Markov chain^{3} is a type of Markov process^{3} that has either discrete state space^{3} or discrete index set^{3}, [...]. Random walks^{1} on integers^{0} and the gambler's ruin problem^{1} are examples^{0} of Markov processes^{3}[...]. Markov chains^{3} have many applications^{0} as statistical models^{1} of real-world processes^{0}, [...]. The algorithm known^{0} as PageRank^{1}, which was originally proposed for the internet search engine Google^{0}, is based on a Markov process^{3}[...]

Figure: An example of an annotated document based on our scoring scheme.

Outline

Experimental Design

- We evaluate Surfking on the heterogeneous collection (DUC + Inspec)
- We randomly split the data into 75% training and 25% testing such all instances originating from the same document remain in the same subset
- We measure the performance of our model by computing Precision, Recall, F1-score and Mean Reciprocal Rank.

$$MRR = \frac{1}{|D|} \sum_{d_t \in D} \frac{1}{r_{d_t}}$$

- We evaluate the top 10 predicted keyphrases returned by the mode

- Baselines:
  - RankingSVM (Jiang et al., 2009)
  - Surfking (binary)
  - SurfKE

Model	Precision	Recall	F1-score	MRR
SurfKE	0.258	0.344	0.282	0.545
RankingSVM	0.234	0.324	0.260	0.534
Surfking (binary)	0.259	0.352	0.287	0.541
Surfking (fine-grained)	**0.272**	**0.361**	**0.298**	**0.556**

Table: Comparison with previous work.

Cross-Domain Performance

- NUS contains 211 research papers and the author-input keyphrases as gold-standard for evaluation.

	NUS			
Approach	P	R	F1	MRR
RankingSVM	0.090	0.388	0.142	0.389
SurfKE	0.101	0.420	0.158	0.405
Surfking	**0.110**	**0.451**	**0.172**	**0.423**

Table: Comparison with previous work in a cross-domain setting.

An Anecdotal Example

Mobile **banking**'s tough sell **Banks** are having to put their **mobile-commerce** projects on hold because the essential technology to make the services usable, in particular **GPRS** (general packet radio service) hasn't become widely available. It is estimated that by the end of 2002, only 5 per cent of adults will have **GPRS phones**. This will have a knock-on effect for other technologies such as clickable icons and multimedia messaging. In fact **banking** via **WAP** (wireless application protocol) has proved to be a frustrating and time-consuming process for the customer. Financial firms' hopes for higher mobile usage are stymied by the fact that improvements to the systems won't happen as fast as they want and the inadequacies of the system go beyond immature technology. Financial services institutions should not wait for customers to become au fait with their **WAP**. Instead they should be the ones "driving the traffic"

Human-input keyphrases: *banking, mobile-commerce, GPRS, wireless application protocol*

Figure: A document from the Inspec collection and the human-annotated

Summary

- We proposed a framework to learn feature representations of phrases that can be used for the task of keyphrase extraction.
- Experimental results show that our supervised model which uses the graph-based features:
    - obtains remarkable improvements in performance over strong baselines.
    - requires a small training set to achieve good performance
- We proposed to reformulate keyphrase extraction problem as a ranking problem using learning to rank anda fine-grain measure of a phrase relevance to the topics of a target document.

Future Work

- Look at other architectures for learning graph-representations that can be leveraged fro keyphrae extraction
- Develop an algorithms to directly optimize the node embeddings using the keyphrase true labels
- Leverage the Listwise LTR approach for keyphrase extraction

Publications

- Wei Jin, Corina Florescu. *A Supervised Keyphrase Extraction System based on Graph Representation Learning* In: **ECIR** 2019.
- Corina Florescu,Cornelia Caragea. *Venue Classification of Research Papers in Scholarly Digital Libraries* In: International Conference on Theory and Practice of Digital Libraries, **TPDL** 2018.
- Corina Florescu, Wei Jin. *Learning Feature Representations for Keyphrase Extraction.* In: **AAAI** 2018.
- Corina Florescu, Cornelia Caragea. *PositionRank: An Unsupervised Approach to Keyphrase Extraction from Scholarly Documents.* In: **ACL** 2017.
- Corina Florescu, Cornelia Caragea. *A Position-Biased PageRank Algorithm for Keyphrase Extraction.* In: **AAAI** 2017.
- Corina Florescu, Cornelia Caragea. *A New Scheme for Scoring Phrases in Unsupervised Keyphrase Extraction.* In: **ECIR** 2017.