Identification and Evaluation of Keyphrases: Fuzzy Set based Scoring and Turing Test Inspired Evaluation

Pashutan Modaresi Stefan Conrad
Heinrich-Heine-University of Düsseldorf
Institute of Computer Science, Düsseldorf, Germany
{modaresi, conrad}@cs.uni-duesseldorf.de

ABSTRACT

Automatic keyphrase extraction aims at extracting a compact representation of a single document, which can be used for numerous applications such as indexing, classification or summarization. Existing keyphrase extraction approaches typically consist of two steps. An extraction step to select the *candidate phrases* using some heuristics and a scoring phase for ranking the extracted candidate phrases based on their importance in the text. Existing approaches to automatic keyphrase extraction mainly define the set of phrases of a document as a crisp set and by scoring and ranking the phrases, they select the keyphrases of the document. In this work we define the set of phrases in a document to be a fuzzy set, and based on the membership values of the phrases, we select the ones with higher membership values as the keyphrases of the document. Moreover we propose a novel evaluation method inspired by the Turing test, which can be used for extractive summarization tasks.

KEYWORDS

Automatic Summarization, Keyphrase Extraction, Keyword Extraction, News Summarization, Fuzzy Sets

1 INTRODUCTION

A Phrase is a group of words acting as a single part of speech. In automatic keyphrase extraction the goal is to automatically extract the most important and significant phrases of a document. Depending on the use case, the extracted phrases can then be used for searching inside a collection of documents, summarizing single or multiple documents [1], document classification [2] and indexing [3]. In some publications the terms *keyword* and *keyphrase* are used interchangeably. It should be noticed that a keyword consists of only one token, but keyphrases have a length equal or greater than one.

Automatic Text Summarization is an active field of research, which aims to deal with the problem of immense available textual information and find a solution to deal with this massive amount of data. Particularly in media branch, summarization plays a significant role and results in great time-savings, while increasing the work efficiency. Extracting keyphrases from a document, can help the reader to instantly attain an overview of the subject matter and contents of a document. Moreover, the extracted keyphrases can be used to extract the most important sentences in the document.

Many approaches to automatic keyphrase extraction define the set of keyphrases of a document as a crisp set. This means that each candidate phrase in the document will either belong to the class *keyphrase* or to the class *not-keyphrase*. We believe that a fuzzy set containing all the phrases of the document with different membership values is a more appropriate mean to represent the phrases of a document. In this way, the decision whether a phrase belongs to the set of keyphrases or not, can be easily made based on its membership value to the fuzzy set.

The process of keyphrase extraction typically consists of two steps. A *candidate selection* phase and a *keyphrase extraction* step. In the first step a set of candidate phrases will be selected from the text. This is normally done using some heuristics to determine the appropriate phrases in the text. Having the set of candidate phrases, a scoring algorithm (either supervised or unsupervised) ranks the candidate phrases regarding their importance in the context of the text and based on a threshold (either user-specified or automatically determined)

returns the ranked list of keyphrases.
Different from existing extraction approaches that introduce a unified algorithm consisting of both phases of keyphrase extraction, in this work we focus on the second step of keyphrase extraction, namely the scoring algorithm. We will assume a given set of candidate phrases that are selected either manually by human annotators or automatically using candidate selection algorithms.
We use an unsupervised fuzzy set approach to automatically score the candidate phrases of a single document. In Section 2 we will shortly introduce the related work. A formal definition of phrases and keyphrases based on the fuzzy sets will be introduced in Section 3. For each phrase in a document we will define a membership degree that determines its significance in the document. The calculation of membership values will be explained in Section 4. To evaluate our algorithm we will introduce a novel evaluation method (Section 5) inspired by the Turing test, and we will represent our evaluation results in Section 6. Finally, we conclude our work in section 7.

2 RELATED WORK

Automatic keyphrase extraction has been applied to various sources like scientific publications [4], web pages [5] and news articles [6]. Usually the algorithms extract a set of phrases in the document as the candidate keyphrases and using a scoring algorithm, the importance of the candidate keyphrases will be determined. For extracting the candidate keyphrases many approaches such as extraction of noun phrases [7], elimination of stop-words to obtain phrases [8] or extraction of words with specific part-of-speech tags [9] have been proposed. We investigate some of these approaches in more detail.
As already mentioned, heuristics are usually applied in order to extract candidate keyphrases. In [10] the document under analysis is first tokenized and from the set of available tokens, stop words will be eliminated. The remaining tokens will form the set of candidate keyphrases. In further steps of the algorithm, the tokens will be merged together to form longer candidate keyphrases. Similar to this approach is the *TextRank* [11] algorithm, which also uses tokens as candidate keyphrases. In TextRank the document is tokenized first and POS tags will be added to the tokens. Using a syntactic filter, specific tokens will be added as vertices to a graph. Finally an edge will be added between vertices that co-occur within a window of N words. In a later phase of the algorithm, longer keyphrases will be constructed based on the results of the ranking algorithm. Similar to the TextRank algorithm is the CollabRank [12] algorithm, which is designed to extract keyphrases from a set of documents (multi-document keyphrase extraction).
In [13] noun phrases are used as candidate keyphrases. Using a base noun phrase skimmer, the algorithm proceeds through a text word-by-word looking for sequences of nouns and adjectives ending with a noun and surrounded by non-noun/adjectives. This approach has the advantage that no detailed parse of sentences is needed. In a well known approach, called *KEA* [14], the extraction of candidate keyphrases is performed in three steps. In the first step, the document text will be cleaned by removing punctuation and similar non-relevant characters. In the second step the candidate keyphrases will be identified from the cleaned text, using several rules, such as:

- Candidate keyphrases have a certain maximum length
- Candidate keyphrases can not be proper names
- Candidate keyphrases can not begin or end with a stop word

In a final step, case folding and stemming will be applied to the selected candidate keyphrases. After extracting candidate keyphrases, they will be assigned a score either using a supervised or an unsupervised approach. In supervised approaches the problem of extraction of keyphrases is usually formulated as a binary classification problem [14]. However, this approach has the disadvantage that the phrases

will either be classified as *keyphrase* or *not-keyphrase* and their importance will not be comparable to each other. In order to overcome this problem supervised algorithms have been designed which are able to learn a phrase rater [15].
As in all supervised algorithms, specific features have to be selected to train the algorithms. Examples of such features are:

- TF-IDF [16]: Frequency of the candidate keyphrase in the given document and its inverse document frequency
- Structural features [17]: Location of keyphrases in the document.
- Syntactic features [18]: Syntactic patterns inside the candidate keyphrases.

Several approaches have also been suggested for unsupervised keyphrase extraction. Graph-based approaches represent the content of the document as a graph and based on the edges connecting the nodes of the graph, determine the importance of the words in the document [11]. Additionally clustering approaches are used to cluster the candidate phrases based on their semantic relationship and in this way it will be ensured that the extracted keyphrases reflect the content of the document [10].
Many other unsupervised phrase ranking algorithms use heuristics based on various features like length and frequency of the phrases [13] or the co-occurrence statistics of the terms inside the phrases [19].

3 PHRASES AS A FUZZY SET

In linguistics, a *phrase* is a group of words acting as a single part of speech. Phrases usually contain a keyword, through which their type and category can be identified. This word is known as the *head* of the phrase. Phrases where their head is a noun are called *noun phrases* [20].
For summarizing news articles through keyphrases, understandable and syntactically correct phrases are needed to be extracted. Identifying the (candidate) phrases in a document is not the main focus of this work and can be fulfilled in various ways. The candidate phrases can either be selected using automatic methods (see section 2) or with the assistance of human annotators. In this work we assume that the candidate phrases are annotated manually by humans. This has the advantage that the quality of the ranking algorithm can be measured independent of the quality of the candidate selection step (assuming that the manually selected phrases are a gold standard and almost perfect).
The selected phrases, form now our set of phrases $\mathcal{K}$ which we define as a fuzzy set:

$$\mathcal{K} = \left\{ \left(x, \mu_{\mathcal{K}}(x)\right) \middle| x \in K,\ \ \mu_{\mathcal{K}}(x) \in [0,1] \right\}. \tag{1}$$

In Equation 1, K is a classical set that contains all the phrases in the document, and $\mu_{\mathcal{K}}(x)$ specifies the grade or degree to which any element $x \in K$ belongs to the fuzzy set $\mathcal{K}$. Larger values of $\mu_{\mathcal{K}}(x)$ indicate higher degrees of membership [21].
Intuitively, the membership value of each phrase will represent its significance in the document. Particularly the difference between K and $\mathcal{K}$ has to be emphasized at this point. The set K is a classical set that contains all the phrases of the document under analysis, independent of the definition of the term *phrase*. This means depending on the definition of the term *phrase*, the set K can contain *noun phrases*, *verb phrases*, a combination of them or even sequences of word intuitively identified by a human annotator as candidate phrases. In contrast to K, the fuzzy set $\mathcal{K}$ has an associated membership value for each phrase inside K. A phrase x with a membership value $\mu_{\mathcal{K}}(x) = 0$ represents a phrase inside the document with the least possible significance and a phrase x' with a membership value $\mu_{\mathcal{K}}(x') = 1$ represents a phrase with the maximum possible significance inside the document. It should be noted that both sets K and $\mathcal{K}$ has the same cardinality.
In order to find the keyphrases more easily using a threshold and also to make the membership values among different documents comparable, we use the normalized membership values of the phrases. For this we compute the *height* of

the fuzzy set $\mathcal{K}$ [21]:

$$h(\mathcal{K}) = \sup_{x \in K} \mu_{\mathcal{K}}(x), \tag{2}$$

and divide $\mu_{\mathcal{K}}(x)$ for all $x \in K$ by height of the fuzzy set $\mathcal{K}$. In this way we obtain the normalized fuzzy set with $h(\mathcal{K}) = 1$.
We define the set of keyphrases of a document to be the α-*cut* of the fuzzy set $\mathcal{K}$, defined as follows:

$$[\mathcal{K}]^{\alpha} = \begin{cases} \{x \in X |\ \mu_{\mathcal{K}}(x) \geq \alpha\} & 0 < \alpha \leq 1 \\ cl(supp(\mathcal{K})) & \alpha = 0 \end{cases}, \tag{3}$$

where $cl(supp(\mathcal{K}))$ is the closure of the support of $\mathcal{K}$ (We will not go into details of this definition as the case $\alpha = 0$ represents the membership value of candidates with the least possible significance inside the document).
Using the above definitions, the problem of finding the keyphrases of a document will be reduced to computing the membership values for the elements of the set K and determining the value of α.

4 MEMBERSHIP FUNCTIONS

In our work we analyze 3 features of the phrases in order to determine their significance within the documents. Length of keyphrases, their token frequency and skip-bigram co-occurrence frequency. Based on these features a final membership value (which is a weighted sum of membership values) will be assigned to each phrase. At the end, phrases with higher membership values will be selected as the keyphrases of the document.

4.1 Token Length of Keyphrases

During our experiments we have noticed that human annotators usually tend to select phrases of token length 2 or 3 as keyphrases. There are of course also cases where keyphrases of token length one or greater than 3 are selected.
In order to force the system to select keyphrases of a specific token length, a size threshold can be hard-coded into the algorithm [22]. However, this will have the disadvantage that some keyphrases will be completely ignored (although they might be significant). As an example consider the case of hard-coding the values 2 and 3 into the algorithm and forcing it to ignore keyphrases of length greater than 3. Given a phrase like *Fifa secretary general Jerome Valcke* with length of 5 tokens, this phrase will not be considered as keyphrase, but probably its consituents like *Fifa* or *Jerome Valcke* will be selected as potential keyphrases. This should be acceptable in an application that uses keyphrases for indexing, but in the case of news summarization, informativeness of keyphrases plays an important role.
In order to assure the before mentioned length property of keyphrases, we use a *generalized bell membership function* which is defined as follows [23]:

$$f_1(x) = bell(x; a, b, c) = \frac{1}{1 + |\frac{x-c}{a}|^{2b}}. \tag{4}$$

In Equation 4, c and a represent the center and width of the membership function respectively. Moreover b is a positive number that controls the slope at the crossover point of the function. To understand the reason behind this function choice, the graph of the function represented in Figure 1 will be helpful. The solid curve represents the bell function $bell(x; 2, 2, 2.5)$. The dotted curves represent the impact of the change of parameter a, and the dashed curves show the impact of the change of the parameter b.

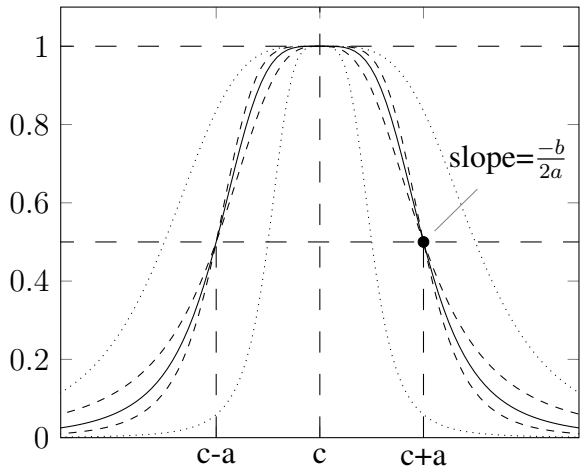

Figure 1. Graph of $bell(x, 2, 2, 2.5)$

By selecting $c = \frac{2+3}{2} = 2.5$ which is the average of the desired length, phrases with length 2

or 3 will be assigned higher membership values than the other phrases. Note that bigger values of b will increase the slope at the crossover point of the function and will cause in greater differences between the membership values of phrases with length 2 or 3, and phrases of other lengths. Due to this property, we recommend small values for b such as $b = 1$ or $b = 1.5$, which will reduce the amount of punishment for very small or very long phrases.

The above mentioned values for the parameters of the bell function are inexact predictions made intuitively based on desired properties of the algorithm. A more accurate approach is needed at this place to compute the parameter values of the bell function. For this we use the method of least squares. We construct a simple data set $D = \{(1, 0.75), (2, 1), (3, 1), (4, 0.75), (5, 0.5)\}$ consisting of (x_i, y_i) instances where x_i represents the token length of a phrase and y_i is its associated membership values. The data set is constructed in a way that phrases with lenghth 2 and 3 get a higher membership value than the other ones. Note that it is assumed that no phrase of length greater than 5 is selected by human annotators (or automatic candidate phrase selectors).

To estimate the parameters of the bell function, the square of residuals has to be minimized.

$$\|r\|^2 = \sum_{i=1}^{5} r_i^2 = \sum_{i=1}^{5} \big[y_i - \frac{1}{1 + |\frac{x_i - c}{a}|^{2b}}\big]^2 \quad (5)$$

The Equation 5 can also be written as:

$$\|r\|^2 = \sum_{i=1}^{5} y_i^2 + \frac{1}{(1 + |\frac{x_i - c}{a}|^{2b})^4} - \frac{2y_i}{(1 + |\frac{x_i - c}{a}|^{2b})^2} \quad (6)$$

In the next step, we compute the partial derivatives of $\|r\|^2$ with respect to the coefficients a, b and c. Equation 7 shows the partial derivative of the function with respect to the parameter a.

$$\frac{\partial \|r\|^2}{\partial a} = \sum_{i=1}^{5} \frac{8ba^{-2b-1}(x_i - c)^2 |x_i - c|^{2b-2}}{(a^{-2b}|x_i - c|^{2b} + 1)^5} \sum_{i=1}^{5} - \frac{8by_i a^{-2b-1}(x_i - c)^2 |x_i - c|^{2b-2}}{(a^{-2b}|x_i - c|^{2b} + 1)^3} \quad (7)$$

The partial derivatives $\frac{\partial \|r\|^2}{\partial b}$ and $\frac{\partial \|r\|^2}{\partial c}$ can be computed analogous to Equation 7. Finally by setting the partial derivatives to zero and solving the resulted linear equations, the values of coefficients will be $a = 2.4494$, $b = 1.1719$ and $c = 2.5111$. As we can see the estimated values are also consistent with our initial predictions. Notice that the parameters a, b and c are not dependent of the document under analysis and have to be set only once.

4.2 Token Frequency

Another property of phrases that plays a role in considering them as a keyphrase is their frequency. Phrases that contain tokens which occur frequently in the document are more likely to be keyphrases.
Let $p = t_1 \dots t_m$ be a phrase consisting of m distinct tokens. We define the frequency of a phrase inside a document as the sum of the frequencies of its constituting tokens. Formally this will be written as:

$$n_p = \sum_{i=1}^{m} n_{t_i}. \quad (8)$$

Note that n_{t_i} is actually the frequency of token t_i inside all available phrases in the document and not the document itself. In this way the effect of stop-words resulting in high frequencies for phrases will be avoided.
In order to model the property that phrases with frequent tokens are more likely than the ones with less frequent tokens to be keyphrases, we use a *sigmoid membership function* defined as follows [21]:

$$f_2(x) = sigmf(x; a', c') = \frac{1}{1 + e^{-a'(x - c')}}. \quad (9)$$

The parameter a' can be specified by the user ($a' = 1$ or $a' = 2$ is recommended). Depending on the sign of the parameter a', the function can be open right or left and thus will be appropriate for representing quantities such as *very frequent* or *very rare*. In order to determine the value of the parameter c', first the frequencies of all phrases (n_p) have to be computed. We compute then the value of c' as the p-Quantil of the calculated frequencies, with $p = 0.9$.
In Figure 2 the solid curve represents the sigmoid function $sigmf(x; 2, 15)$. The dotted curves represent the impact of the change of parameter c' and the dashed curves show the impact of the change of the parameter a'.

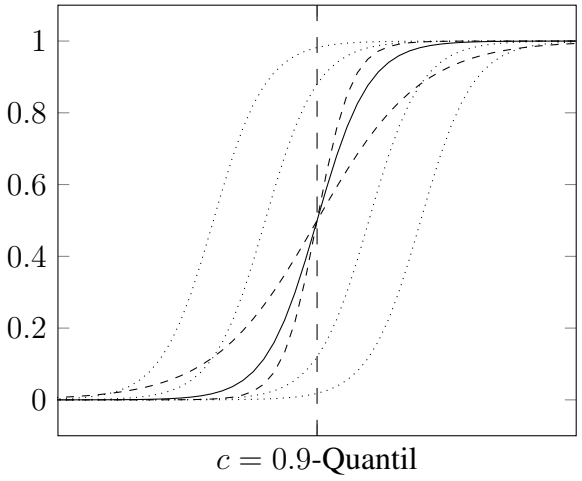

Figure 2. Graph of $sigmf(x, 2, 15)$

The parameter c' represents an absolute number and depending on the document under analysis can take various values. This makes the process of learning this parameter more difficult and document-dependent. In order to solve this problem we perform a zero-one normalization to transform the token frequency values into the range $[0, 1]$. Let $X = \{x_1, \ldots, x_n\}$ be the token frequencies of the n phrases in a document d. Each value x_i can be transformed as follows:

$$x_{i_{norm}} = \frac{x_i - min(X)}{max(X) - min(X)} \qquad (10)$$

Similar to the previous section we construct a data set D to learn the parameters a' and c' (See Table 1).
The dataset represented in Table 1 is designed in a way that the normalized token frequencies

Table 1. Dataset to learn parameters of the sigmoid function

index	x	y
1	0.1	0.05
2	0.2	0.1
3	0.3	0.15
4	0.4	0.3
5	0.5	0.35
6	0.6	0.4
7	0.7	0.8
8	0.8	0.9
9	0.9	1.0
10	1.0	1.0

closer to 1 will be assigned very high membership values and the ones closer to zero will be mapped to very low membership values.
Using the least square method, we minimize the following function:

$$\|r\|^2 = \sum_{i=1}^{10} y_i - \frac{1}{1 + e^{-a'(x_{i_{norm}} - c')}} \qquad (11)$$

By setting the partial derivatives $\frac{\partial \|r\|^2}{\partial a'}$ and $\frac{\partial \|r\|^2}{\partial c'}$ to zero and solving the resulted linear equations the values of coefficients will be $a = 7.6032$ and $c' = 0.5674$. Notice that the parameter values gained using the quantile approach are different from the values estimated using the least square method. This difference is expected and is because of the fact that in the case of quantile approach, absolute values of token frequencies are used and in the case of the least square method, normalized token frequencies are needed. Our experimental results show that both approaches return comparable results.
Notice that we did not consider the effect of length of phrases in the definition of the membership function. This will of course result in higher membership values for longer phrases. However this effect will be canceled by means of the generalized bell function which we already defined in section 4.1.

4.3 Skip-Bigram Co-Occurrence Frequency

Another feature from which we believe to play a significant role in identifying the keyphrases

of a document is the *skip-bigram co-occurrence frequency*. Assuming a phrase $p = t_1 \dots t_m$ consisting of m tokens, *skip-bigrams* are any pair of tokens inside the phrase, with arbitrary gaps [24]. As an example the phrase $p = t_1 t_2 t_3$ will consist of the following skip-bigrams: $(t_1, t_2), (t_1, t_3), (t_2, t_3)$. The total number of skip-bigrams inside a phrase of length m can be computed as the *2-combination* of the set of its tokens which is equal to $\binom{m}{2} = \frac{m!}{(m-2)!2!}$.
For each skip-bigram in the document we calculate the frequency of it among the set of skip-bigrams of available phrases. Finally, the skip-bigram co-occurrence frequency of a phrase will be computed as the sum of the frequencies of all its constituting skip-bigrams. This will be formally written as:

$$n_p^{skip} = \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} n_{t_i,t_j}, \qquad (12)$$

where n_{t_i,t_j} denotes the frequency of the skip-bigram (t_i, t_j). After computing the n_p^{skip} values for all phrases in the document, we again apply the sigmoid membership function to the computed values. As in the previous section, we select the parameter c'' to be the p-Quantil of the n_p^{skip} values with $p = 0.9$. We also recommend the parameter a'' to be equal to 1 or 2. The skip-bigram co-occurrence frequency of a phrase x will be denoted as $f_3(x)$.
Similar to the calculations performed by the token frequency feature, the parameters of the sigmoid membership function can be computed using the least square method. For this the n_p^{skip} values have to be normalized using the zero-one normalization method. We use the same data set as the one used for token frequency feature and calculate the parameter values as $a = 7.6032$ and $c = 0.5674$.

4.4 Final Membership Value Computation

As already mentioned in Section 3, our goal was to assign a membership value to each candidate phrase in the document. We have splitted this problem into 3 sub-problems and using 3 features, namely the phrase length f_1, token frequency f_2 and skip-bigrams co-occurrence frequency f_3, 3 membership functions have been defined. Finally, we define the membership value of a phrase x inside the set K to be the weighted sum of the 3 introduced membership functions:

$$\begin{aligned}\mu_K(x) = \omega_1 \cdot bell(f_1(x); a, b, c) \\ + \omega_2 \cdot sigmf(f_2(x); a', c') \\ + \omega_3 \cdot sigmf(f_3(x); a'', c''), \end{aligned} \qquad (13)$$

with $\sum_{i=1}^{3} w_i = 1$.
In our experiments we use $w_1 = 0.5$, $w_2 = 0.25$ and $w_3 = 0.25$. The keyphrases from the document can now be selected using the Equation 3. For larger values of α the algorithm will select the phrases which are most likely to be a keyphrase in a document. It should be also clear that by setting a higher value for α the number of extracted keyphrases from the document will be decreased and vice versa.

5 IMITATION GAME INSPIRED EVALUATION

In this section we introduce our evaluation method inspired by the imitation game suggested by Alan Turing in 1950 to decide whether a machine is intelligent or not [25].
Assuming a collection $D = \{d_1, \dots, d_n\}$ of n documents, our goal is to compare the automatically ranked keyphrases to the ones ranked by humans. For each document d_i with $1 \leq i \leq n$, let $K_i^{\mathcal{M}} = \{x_{i1}, \dots, x_{im}\}$ and $K_i^{\mathcal{A}} = \{x'_{i1}, \dots, x'_{im}\}$ be the set of manually and automatically ranked keyphrases respectively. Note that the set of automatically and manually ranked keyphrases, both have the same cardinality, namely m. The sets $K_i^{\mathcal{M}}$ and $K_i^{\mathcal{A}}$ posses the following properties:

1. Reflexivity
   $x_{ij} \leq x_{ij}$ for all $x_{ij} \in K_i^{\mathcal{M}}$
   $x'_{ij} \leq x'_{ij}$ for all $x'_{ij} \in K_i^{\mathcal{A}}$
2. Transivity
   $x_{ij} \leq x_{ik}$ and $x_{ik} \leq x_{is}$ implies $x_{ij} \leq x_{is}$
   $x'_{ij} \leq x'_{ik}$ and $x'_{ik} \leq x'_{is}$ implies $x'_{ij} \leq x_{is}$

The evaluation process consists of n iterations, and two human raters $\mathcal{R}_1$ and $\mathcal{R}_2$. In

each iteration a document d_i with its associated keyphrases $K_i^{\mathcal{M}}$ and $K_i^{\mathcal{A}}$ is shown to the raters (The raters are not told which of the keyphrases are human-ranked and which are machine-ranked). The raters are asked to classify the keyphrases either to the class *human-ranked (HR)* or to the class *machine-ranked (MR)*.

In this way the set of labels of the classification problem will be $C = \{\text{human-ranked}, \text{machine-ranked}\}$ and for each set of keyphrases K_i the raters have to determine its label $c(K_i)$. Note that the raters are not allowed to assign the same label for both sets of keyphrases. This means that assuming the sets $K_i^{\mathcal{A}}$ and $K_i^{\mathcal{M}}$, we will have $c(K_i^{\mathcal{M}}) \neq c(K_i^{\mathcal{A}})$. Finally, for each rater a confusion matrix will be constructed (see Table 2).

Table 2. Confusion Matrix of a Rater

		Predicted Labels		
		HR	MR	Total
Actual Labels	HR	A	B	$A+B$
	MR	C	D	$C+D$
	Total	$A+C$	$B+D$	$2n$

Notice that on the bottom right corner of the Table 2, the total number of classifications is written as $2n$ and not n. For each document d_i a rater will be encountered with two sets of keyphrases, namely $K_i^{\mathcal{M}}$ and $K_i^{\mathcal{A}}$. The rater will classify one of the sets as *machine-ranked* and the other one as *human-ranked*. This means that in each iteration a rater performs two classifications. As we have a total number of n documents, the total number of classifications at the end will be equal to $2n$.

In order to assess the agreement on the classification task, we use the *Cohen's Kappa Coefficient* [26] which is a mean to measure the inter-rater agreement and is defined as follows:

$$\kappa = \frac{Pr(a) - Pr(e)}{1 - Pr(e)}. \tag{14}$$

In Equation 14, $Pr(a)$ denotes the observed percentage of agreement and $Pr(e)$ is the expected percentage of agreement. We consider $\kappa \geq 0.60$ to be a good amount of agreement between the raters. For the case $\kappa < 0.60$ more documents have to be added to the collection, until the desired value is reached.

Finally, we compute the average accuracy of the raters $\tilde{\alpha}_{\mathcal{H}}$ as follows:

$$\tilde{\alpha}_{\mathcal{H}} = \frac{1}{4n} \sum_{i=1}^{2} A_i + D_i. \tag{15}$$

In Equation 15, A_i represents the number of times that rater $\mathcal{R}_i$ correctly classified a human-ranked set and D_i denotes the number of times that a machine-ranked set has been correctly classified by $\mathcal{R}_i$.

Based on the average accuracy of the raters, the accuracy of the keyphrase scoring algorithm can be determined. Intuitively this means that in cases where a rater classifies a set of keyphrases as human-ranked, but the set is actually machine-ranked, this would mean that the keyphrase extraction algorithm was successful in imitating the humans and has ranked keyphrases, even better than the human-ranked ones. Based on this idea we define the accuracy of the keyphrase scoring algorithm $\tilde{\alpha}_{\mathcal{M}}$ to be the *misclassification rate* of the human raters:

$$\tilde{\alpha}_M = 1 - \tilde{\alpha}_{\mathcal{H}}. \tag{16}$$

The lower the average accuracy of human raters, the higher the accuracy of the algorithm will be.

6 EVALUATION RESULTS

For our experiments we have collected 30 English news articles from *BBC News* and for each document inside our collection we have manually extracted the top 10 keyphrases from the documents. The extraction process has been performed by two annotators, and from the extracted keyphrases of each annotator, an intersection set of size 10 has been selected. In this way we assure the reliability of the extracted keyphrases. Additionally we used our algorithm to rank the top 10 manually extracted keyphrases from the documents.

For each document d_i, with $1 \leq i \leq 30$ we asked the raters to decide which element of the pair $(K_i^{\mathcal{H}}, K_i^{\mathcal{M}})$ is machine-ranked and which one is human-ranked (the actual labels of

the keyphrases are of course not shown to the raters).
The confusion matrices of the raters $\mathcal{R}_1$ and $\mathcal{R}_2$ can be seen in Table 3 and Table 4 respectively.

Table 3. Confusion Matrix $\mathcal{R}_1$

		Predicted Labels		
		HR	MR	Total
Actual Labels	HR	12	18	30
	MR	18	12	30
	Total	30	30	60

Table 4. Confusion Matrix of $\mathcal{R}_2$

		Predicted Labels		
		HR	MR	Total
Actual Labels	HR	47	53	30
	MR	53	47	30
	Total	30	30	60

To compute the *Cohen's Kappa Coefficient*, we collected the following information about the ratings:

1. Number of times that $\mathcal{R}_1$ and $\mathcal{R}_2$ classified the same set of keyphrases as human-ranked = 24
2. Number of time that $\mathcal{R}_1$ and $\mathcal{R}_2$ classified the same set of keyphrases as machine-ranked = 24
3. Number of times that $\mathcal{R}_1$ classified a set of keyphrases as human-ranked but $\mathcal{R}_2$ classified the same set as machine-ranked = 6
4. Number of times that $\mathcal{R}_2$ classified a set of keyphrases as human-ranked but $\mathcal{R}_1$ classified the same set as machine-ranked = 6

In total the number of observed agreements will be 48 (80% of the observations) and the number of expected agreements by chance is 30 (50% of the observations). Thus for *Cohen's Kappa Coefficient* we will have $\kappa = 0.6$ which is considered to be *good*.
Using the information contained in Table 3 and Table 4 we also compute the average accuracy of the raters which is $\tilde{\alpha}_{\mathcal{H}} = 0.4\overline{3}$. Finally, the accuracy of the algorithm will be $\tilde{\alpha}_{\mathcal{M}} = 1 - 0.4\overline{3} = 0.5\overline{6}$.

Act now on climate change or see costs soar, White House says
(Reuters) - Putting off expensive measures to curb climate change will only cost the United States more in the long run, the White House said on Tuesday in a report meant to bolster a series of actions President Barack Obama has proposed to address global warming.
"Each decade we delay acting results in an added cost of dealing with the problem of an extra 40 percent," said Jason Furman, chairman of Obama's Council of Economic Advisers.
"We know way more than enough to justify acting today," Furman told reporters.
The report drew its conclusions from 16 economic studies that modeled the costs of climate change. It was released as the U.S. Environmental Protection Agency holds public hearings on its plan to cut carbon emissions from power plants - the centerpiece of Obama's climate action plan.
Business groups have said the EPA's plan would hurt jobs in the coal sector and harm the U.S. economy.
The White House and environmental groups have pushed back against that argument.

Figure 3. A snippet of a news article

As an example for the output of our algorithm we extracted the top 10 keyphrase of a news article[1] from *Reuters*. A snippet of the news article can be seen in Figure 3. The top 10 automatically ranked keyphrases can be seen in Figure 4.

7 CONCLUSIONS

In this work we have introduced a fuzzy set approach to rank keyphrases from news articles. We defined several fuzzy membership functions to calculate the significance of candidate phrase and by means of a weighted membership func-

[1] http://www.reuters.com/article/2014/07/29/us-usa-climatechange-idUSKBN0FY0V820140729

energy and climate change, White House and environmental groups, natural gas transmission and distribution system, climate costs, White House, former New York City Mayor Michael Bloomberg, agriculture and food production, U.S. Environmental Protection Agency, Energy Secretary Ernie Moniz, methane emissions

Figure 4. Extracted keyphrases from the news article

tion, determined the final importance of the them. Finally we introduced a novel evaluation approach inspired by the Turing test.
Using the introduced approach we were able to partly imitate the humans in the way they rank keyphrases in documents. This led to the situation where in some cases our raters were unable to differentiate between the human-ranked and machine-ranked keyphrases. Also in many cases the machine-ranked keyphrases were wrongly classified by the judges as human-ranked keyphrases which is an indicator of the acceptable performance of our algorithm.
For future work we intend to perform an in-depth analysis of our evaluation method and generalize it to further types of summarization such as *sentence extraction* or even *abstractive summarization*.

REFERENCES

[1] Y. Zhang, E. Milios, and N. Zincir-heywood, "A comparative study on key phrase extraction methods in automatic web site summarization," *Journal of Digital Information Management, Special Issue on Web Information Retrieval*, 2007.

[2] L. Dolamic and C. Boyer, "Key-phrase based classification of public health web pages.," in *MedInfo* (C. U. Lehmann, E. Ammenwerth, and C. Nhr, eds.), vol. 192 of *Studies in Health Technology and Informatics*, p. 1133, IOS Press, 2013.

[3] N. Erbs, I. Gurevych, and M. Rittberger, "Bringing order to digital libraries: From keyphrase extraction to index term assignment.," *D-Lib Magazine*, vol. 19, no. 9/10, 2013.

[4] S. N. Kim, O. Medelyan, M.-Y. Kan, and T. Baldwin, "Semeval-2010 task 5: Automatic keyphrase extraction from scientific articles," in *Proceedings of the 5th International Workshop on Semantic Evaluation*, SemEval '10, (Stroudsburg, PA, USA), pp. 21–26, Association for Computational Linguistics, 2010.

[5] M. Grineva, M. Grinev, and D. Lizorkin, "Extracting key terms from noisy and multitheme documents," in *Proceedings of the 18th International Conference on World Wide Web*, WWW '09, (New York, NY, USA), pp. 661–670, ACM, 2009.

[6] X. Wan and J. Xiao, "Single document keyphrase extraction using neighborhood knowledge.," in *AAAI* (D. Fox and C. P. Gomes, eds.), pp. 855–860, AAAI Press, 2008.

[7] K. Barker and N. Cornacchia, "Using noun phrase heads to extract document keyphrases," in *Advances in Artificial Intelligence* (H. Hamilton, ed.), vol. 1822 of *Lecture Notes in Computer Science*, pp. 40–52, Springer Berlin Heidelberg, 2000.

[8] S. Rose, W. Cowley, V. Crow, and N. Cramer, "Rapid automatic keyword extraction for information retrieval and analysis," Mar. 6 2012. US Patent 8,131,735.

[9] F. Liu, D. Pennell, F. Liu, and Y. Liu, "Unsupervised approaches for automatic keyword extraction using meeting transcripts," in *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '09, (Stroudsburg, PA, USA), pp. 620–628, Association for Computational Linguistics, 2009.

[10] Z. Liu, P. Li, Y. Zheng, and M. Sun, "Clustering to find exemplar terms for

keyphrase extraction," in *In Proceedings of EMNLP*, pp. 257–266, 2009.

[11] R. Mihalcea and P. Tarau, "TextRank: Bringing order into texts," in *Proceedings of EMNLP-04and the 2004 Conference on Empirical Methods in Natural Language Processing*, July 2004.

[12] X. Wan and J. Xiao, "Collabrank: Towards a collaborative approach to single-document keyphrase extraction," in *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, (Manchester, UK), pp. 969–976, Coling 2008 Organizing Committee, August 2008.

[13] K. Barker and N. Cornacchia, "Using noun phrase heads to extract document keyphrases," in *Proceedings of the 13th Biennial Conference of the Canadian Society on Computational Studies of Intelligence: Advances in Artificial Intelligence*, AI '00, (London, UK, UK), pp. 40–52, Springer-Verlag, 2000.

[14] I. H. Witten, G. W. Paynter, E. Frank, C. Gutwin, and C. G. Nevill-Manning, "Kea: Practical automatic keyphrase extraction," in *Proceedings of the Fourth ACM Conference on Digital Libraries*, DL '99, (New York, NY, USA), pp. 254–255, ACM, 1999.

[15] D. Kelleher and S. Luz, "Automatic hypertext keyphrase detection.," in *IJCAI* (L. P. Kaelbling and A. Saffiotti, eds.), pp. 1608–1609, Professional Book Center, 2005.

[16] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Inf. Process. Manage.*, vol. 24, pp. 513–523, Aug. 1988.

[17] T. D. Nguyen and M. yen Kan, "Keyphrase extraction in scientific publications," in *In Proc. of International Conference on Asian Digital Libraries (ICADL 07*, pp. 317–326, Springer, 2007.

[18] W.-t. Yih, J. Goodman, and V. R. Carvalho, "Finding advertising keywords on web pages," in *Proceedings of the 15th International Conference on World Wide Web*, WWW '06, (New York, NY, USA), pp. 213–222, ACM, 2006.

[19] Y. Matsuo and M. Ishizuka, "Keyword extraction from a single document using word co-occurrence statistical information," *International Journal on Artificial Intelligence Tools*, vol. 13, p. 2004, 2004.

[20] P. H. Matthews, *The concise Oxford dictionary of linguistics*. Oxford paperback reference, Oxford, England, New York: Oxford University Press, 2007.

[21] G. J. Klir and B. Yuan, *Fuzzy Sets and Fuzzy Logic: Theory and Applications*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1995.

[22] L. Wang and F. Li, "Sjtultlab: Chunk based method for keyphrase extraction," in *Proceedings of the 5th International Workshop on Semantic Evaluation*, SemEval '10, (Stroudsburg, PA, USA), pp. 158–161, Association for Computational Linguistics, 2010.

[23] G. Feng, *Analysis and Synthesis of Fuzzy Control Systems: A Model-Based Approach*. Boca Raton, FL, USA: CRC Press, Inc., 1st ed., 2010.

[24] S. Li, Y. Zhang, W. W. 0013, and C. Wang, "Using proximity in query focused multi-document extractive summarization.," in *ICCPOL* (W. Li and D. M. Aliod, eds.), vol. 5459 of *Lecture Notes in Computer Science*, pp. 179–188, Springer, 2009.

[25] A. M. Turing, "Computing machinery and intelligence," *Mind*, vol. 59, no. 236, pp. 433–460, 1950.

[26] J. Carletta, "Assessing agreement on classification tasks: The kappa statistic," *Comput. Linguist.*, vol. 22, pp. 249–254, June 1996.