Semantic Documents Relatedness using Concept Graph Representation

Yuan Ni
IBM Research, China
niyuan@cn.ibm.com

Qiong Kai Xu
IBM Research, China
xuqkai@cn.ibm.com

Feng Cao
IBM Research, China
caofeng@cn.ibm.com

Yosi Mass
IBM Research, Haifa
yosimass@il.ibm.com

Dafna Sheinwald
IBM Research, Haifa
dafna@il.ibm.com

Hui Jia Zhu
IBM Research, China
zhuhuij@cn.ibm.com

Shao Sheng Cao
Xidian University
shelsoncao@gmail.com

ABSTRACT

We deal with the problem of document representation for the task of measuring semantic relatedness between documents. A document is represented as a compact *concept graph* where nodes represent concepts extracted from the document through references to entities in a knowledge base such as DBpedia. Edges represent the semantic and structural relationships among the concepts. Several methods are presented to measure the strength of those relationships. Concepts are weighted through the concept graph using closeness centrality measure which reflects their relevance to the aspects of the document. A novel similarity measure between two concept graphs is presented. The similarity measure first represents concepts as continuous vectors by means of neural networks. Second, the continuous vectors are used to accumulate pairwise similarity between pairs of concepts while considering their assigned weights. We evaluate our method on a standard benchmark for document similarity. Our method outperforms state-of-the-art methods including ESA (Explicit Semantic Annotation) while our concept graphs are much smaller than the concept vectors generated by ESA. Moreover, we show that by combining our concept graph with ESA, we obtain an even further improvement.

Categories and Subject Descriptors

H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing; I.2.4 [**Artificial Intelligence**]: Semantic Networks; I.2.7 [**Artificial Intelligence**]: Text Analysis

WSDM'16, February 22–25, 2016, San Francisco, CA, USA.
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-3716-8/16/02 ...$15.00.
DOI: http://dx.doi.org/10.1145/2835776.2835801 .

General Terms

Algorithms, Performance

Keywords

Document Representation, Document Semantic Similarity, DBpedia, Graph Model, Neural Network

1. INTRODUCTION

Semantic relatedness, or similarity between documents plays an important role in many textual applications such as information retrieval, document classification and clustering, question answering and more. Measurement of semantic relatedness comprises two constituents: an effective representation of documents, and a similarity measure between documents in terms of their respective representations.

Most document representations map the documents to vectors of a fixed length, aiming to map semantically related documents to close vectors in the vector space. The simplest representation of these is the *bag-of-words*, where a document is represented as a vector of frequencies organized with respect to a vocabulary of words. Component i of the vector reflects the frequency, in the represented document, of the i-th word of the vocabulary. Bag-of-words only represents the syntax of the words and their frequencies. It does not address multiple meanings of same word or synonymy of words. It also totally ignores the order of the words in the document.

Latent topic models, such as *Latent Semantic Analysis* [5], *Latent Dirichlet Allocation* [2], or *Word2Vec* [11], map words and documents to vectors whose components represent latent topics. The (configurable) length of those vectors is much smaller than with bag-of-words. The effectiveness of this representation for indication of semantic relatedness of documents was experimentally verified. The latent topics, however, are hard to interpret, they are not as intuitive as the components of the bag-of-words vectors.

The growing popularity of Semantic Web, Wikipedia, DBpedia and other universal knowledge bases, brought a new type of representing vectors, where the i-th component reflects the relative weight, or relevance of the i-th concept of the knowledge base in the represented document. Thus, the length of these concept vectors equals the number of concepts in the knowledge base, which could be in the millions. Nevertheless, the simple correspondence between components and concepts makes these vectors intuitive, easier for human interpretation. Gabrilovich and Markovitch initiated this direction with their ESA [7].

The representations and similarity measurements described above have some shortcomings. The bag-of-words model lacks reflection of the semantics of the text. The topic models (both latent and explicit) do not consider the structure and relationships among the topics or concepts used for the representation.

In this paper we propose a novel document representation and a measurement of similarity, that combine the advantages of them all. Specifically, our method i) uses topic models that are based on explicit concepts ii) considers the relationships among the concepts and iii) uses neural network based methods to represent concepts (as opposed to words) as continuous vectors.

For document representation we build a graph whose nodes are explicit concepts from a knowledge base. Similarly to Schuhmacher and Ponzetto [16], we extract the concepts directly from the document text using a mention detection tool such as SpotLight [4] or TagME [6]. However, there are several differences from [16]. First, we observe that some concepts may be closely related to the main aspects of the document while others may drift. Consider, for example, the text segment:

*The **study** shows an increase in **forced abortions, female infanticide***

Where concepts **study**, **forced abortions** and **female infanticide** are detected. Obviously, concept **study** is not as relevant to the text's aspect as the other two concepts.

Thus we assign weights to nodes in the graph as described below, while [16] only assigns weights to edges in the graph. The second major difference from [16] is that our similarity measure exploits the assigned weights with an extension of Word2Vec [11] to represent concepts as continuous vectors, while the similarity in [11] is based on Graph Edit Distance that considers only transitions between nodes in the graph.

To capture the relative importance, or weight, of the detected concepts, we arrange them as nodes in a *concept graph*, and use the knowledge base to assign weights to edges between each pair of concepts. We consider different types of relationships and structural links among the concepts and their environment in the knowledge base. Similarly to [9], we argue that concepts with higher coherence are more important to the aspects of the document. To evaluate the coherence of each concept with other concepts, we use the concept's level of centrality, which measures its closeness, through the weighted edges of the concept graph, to the rest of the concepts. We take the concept's centrality level to be its weight in the concept graph.

To evaluate semantic relatedness between two documents, we measure the similarity between their concept graphs as the weighted sum of contributions from all pairs of nodes, one from each graph, of their pairwise similarity. The weights are the weights of the nodes in their respective concept graphs. We suggest several pairwise similarity measures, one of which is a novel measure, for which we propose *Concept2Vec* that expands upon Word2Vec [11] to represent a concept as a continuous vector of 200 dimensions.

We evaluate our document representation and similarity measures on LP50 [10], a standard benchmark for document similarity. We show that our method outperforms the state-of-art method ESA, while our concept graphs are much smaller than the concept vectors generated by ESA. Moreover, we show that by combining our concept graph with ESA, we exceed each method separately, thus outperforming all other known systems.

To summarize, our contributions are the following:

- We propose a document representation as a concept graph whose nodes and edges are weighted. In its weighted edges, the concept graph reflects the strength of relationships among concepts and their environment. We propose several methods to weight the edges, and propose the centrality measure to assign weights to the nodes, which indicate the importance of the different concepts to the aspects of the document.
- We propose Concept2Vec for representing concepts as continuous vectors using neural networks that extend Word2Vec from words to concepts.
- We propose a measure of similarity between concept graphs.
- We evaluate our method on a document similarity benchmark and show that it outperforms other methods. We conduct extensive experiments to verify the effectiveness of each of the components of our method.

The rest of the paper is organized as follows. Section 2 presents related work. Section 3 provides an overview of our methods. Section 4 describes the detailed techniques to build the concept graph and in Section 5 we describe how we assign weights to the concepts in the concept graph. Section 6 introduces the Concept2Vec and the similarity measure between two concept graphs. Section 7 evaluates experimental results and we conclude our paper in Section 8.

2. RELATED WORK

One of the first, plain representations introduced is the *bag-of-words*, where the document is represented by the multiset (bag) of the words making up its text. This gives rise to the *Vector Space Model* (VSM) [15], where all documents are represented as vectors of same number of components, being the size of the underlying word vocabulary. Component i of the vector representing document d reflects the frequency of the i-th vocabulary word in d. By VSM, semantically related documents are represented by close vectors, or points, in the vector space. Specifically, similarity between documents is measured by the angle (or the cosine of the angle) between their representing vectors. Optional tunings of the vector's entries, like *idf* weighting, or normalization by document length, are often used, having been shown to enhance accuracy of indication of semantic similarity. Bag-of-words does not address polysemy (the same word can have multiple meanings) and synonymy (two words can represent the same concept), and it totally ignores the order of the words in the document. It is thus weak in reflecting an overall semantic meaning of the document. Subsequent, more intricate variations of VSM [18][14] aim at alleviating this problem through employing an extended vocabulary of pairs of words, or even larger sets, building on the distributional hypothesis by which words that occur in similar contexts tend to have similar meanings.

Also building on same hypothesis, latent topic models such as *Latent Semantic Analysis* (LSA) [5], *Latent Dirichlet Allocation* (LDA) [2], or *Word2Vec* [11], establish associations between words, or terms that occur in similar contexts, associations that are extracted from example corpora. The methods differ in the way they derive these associations from the example data, ranging from mathematical operations on matrices formed from the bag-of-words vectors, to neural networks, but similar in the expression of the derived associations through the representation of words and documents as (continuous) vectors, of a much lower (configurable) dimension than the original bag-of-words vectors, such that vectors representing semantically similar words or documents are close to one another in the vector space, as can be measured by the cosine distance between them. Latent topic models circumvent synonymy since they map the documents into a vector space of (latent) topics instead of using words. These latent topics, however, are hard to interpret, they are not as intuitive as the original bag-of-words vectors.

With the proliferation of Semantic Web, Wikipedia, DBpedia and other universal knowledge bases, which are continually expanded by the general crowd, and abundantly referenced as a source of high dimensional space of fine grained natural concepts, a new form of vector representation evolved, which reflects the knowledge base concepts that are relevant to the represented document. The semantic meanings of a document, as expressed through these concepts, is easier for human interpretation. *Explicit Semantic Analysis* (ESA) by Gabrilovich and Markovitch [7] was pioneering in this direction, expressing of the meaning of any text in terms of a weighted vector of Wikipedia concepts. Wikipedia contains an article for each of its concepts, whose title is that concept. ESA quantifies the strength of association $assoc(w, c)$ of word w with concept c as the TFIDF weight of word w in the Wikipedia article titled c. The strength of association $assoc(T, c)$ of text T with Wikipedia concept c is defined as the sum, over the words $\{w_i\}$ of T, of $assoc(w_i, c)$ multiplied by the TFIDF weight of w_i in T. These $assoc(T, c)$ values, organized in a vector whose length is the number of Wikipedia concepts (a few millions), form the ESA representation of T. The Wikipedia concepts that make up the ESA representation have explicit semantic meanings, and are intuitive for human understanding, as opposed to the obscure topics of the above latent topic models. Similarly to the previous methods, semantic relatedness of documents is measured by the cosine of the angle between their respective concept vectors.

While ESA totally ignores Wikipedia's wikilinks, which are its inter-article links, Yeh et al. [19] employ a random walk on the Wikipedia graph, whose nodes are Wikipedia's articles, or concepts, and edges – its wikilinks. The walk starts with a distribution of mass over the nodes, a distribution derived from the given text, either as the end result of ESA on the given text, or through detection of mentions of Wikipedia concepts in the text, by existing or especially developed tools. The stationary distribution converged to by the random walk yields the representation of the text as a vector whose length equals the number of Wikipedia articles. Semantic relatedness between two texts is computed as with ESA - by the cosine similarity of their concept vector representations.

Unlike ESA that uses a large number of concepts, methods like Schuhmacher and Ponzetto [16] only reference a small number of DBPedia concepts, relying, instead, on their explored inter-relatedness. Given a text, mentions of DBPedia concepts within it are first identified using existing tools, such as SpotLight [4] or TagME [6]. A graph is then generated whose nodes are the identified concepts, and the concepts which are connected to them through a semantic relation found in DBpedia, and whose edges are these applied DBpedia semantic relations. The edges are further weighted using information theoretic measures, independent of the given text, aiming to capture the degree of associativity between the concepts connected by each edge. Semantic relatedness between two documents is measured by a tailored variation of the graph edit distance (GED) between their respective graph representations, a variation which only considers distance between nodes. Node distance, in turn, is computed along the paths of DBpedia's semantic relations, weighted as suggested above. For similarity between documents, the nodes themselves are not weighted, namely, a pair of nodes (one from each graph) contributes the distance between them to the total edit distance, without any notion of weight, or significance to the analyzed text, of each node relative to the other nodes in its graph. In contrast, we assign weights to nodes based on their importance in the text and our similarity measure combines the weights with an extension of Word2Vec [11] for concepts' representation.

Huang et al [9] continue a line of works that represent a document using a bag of words and concepts. They define several features that capture concept centrality. Some of their features consider the local concept centrality of concepts among other concepts in the same document. For document similarity they define other features that consider the relative concept centrality of concepts in one document with respect to concepts in the other document. Altogether, a total of 17 features are specified. They use machine learning to learn the weights of the features. Our method uses a similar notion of centrality, although slightly different in the details, and fewer features. We believe that our results can improve with additional features, and with tuning per subject domain.

3. OVERVIEW

In this section, we overview our concept graph and its generation from a given document. Techniques are specified in detail in section 4.2.

The key idea of document representation by a concept graph is to link the concepts in the document text to entities of a knowledge base, and to explore the structural and semantic information among them in the knowledge base, to create a graph for these entities. Based on the observation that the core aspects of a document should be a set of closely related concepts, we measure the centrality, over the graph, of each concept to obtain its weight.

Figure 1 depicts the architecture of our method. First, given the input document, we use an existing mention detection tool such as Spotlight [4] or TagMe [6] to detect the mentions in the document text. A mention is a term/phrase in the text that corresponds to a defined concept in the knowledge base. In this paper, we use DBpedia as our knowledge base. The English version of DBpedia 2014 comprises 4.58 million concepts, each corresponds to a Wikipedia article. These are annotated with 583 million facts. Given

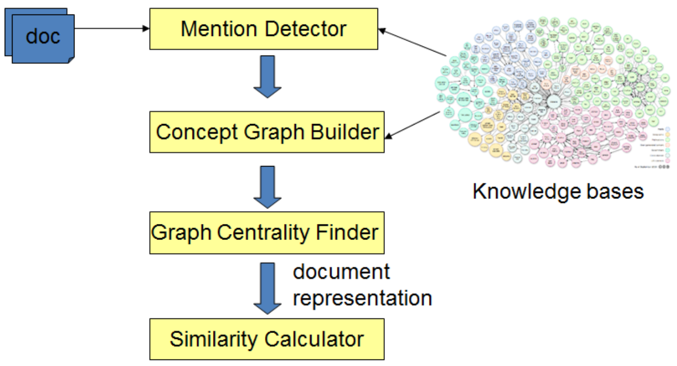

Figure 1: Architecture of our method

the following document text, for example, the tool will detect four mentions and their corresponding concepts (shown in brackets).

Gambling [http://dbpedia.org/resource/Gambling] *increases* ***aggregate demand*** [http://dbpedia.org/resource/Aggregate_demand] *for* ***goods and services*** [http://dbpedia.org/resource/Goods_and_Services] *in the* ***economy*** [http://dbpedia.org/resource/Economy]

The nodes of the concept graph are the detected concepts in the given document. We measure the strength of association between each pair of concepts, based, as follows, on their features and their relationship in the knowledge base. Any pair of concepts with a non-zero association between them is connected by an edge in the concept graph, whose weight is taken to be the strength of that association. The inter-concept association is calculated as a combination of three types of associations, each reflecting the extent of similarity, or relatedness of the concepts in the scope of one feature:

- Context association: relative frequency of occurrence of both concepts in same contexts.
- Category association: how close the concepts are in the knowledge base taxonomy of categories
- Structure association: the related structural information from the knowledge base

Once the edges are determined, the centrality of each node in the concept graph is computed, and set to be the weight of the node.

Finally we define several measures of similarity between two concept graphs which aim to reflect the semantic relatedness of the documents they represent.

4. CONCEPT GRAPH GENERATION

We start with a brief background of the DBpedia knowledge base and continue to a detailed description of the generation of our concept graph.

4.1 DBpedia Data

As aforementioned, our method relies on the semantic association and structural relationship from a knowledge base. We choose DBpedia [1] as our knowledge base due to its large coverage and variety of relationship types on both ontology level and instance level. The proposed approach, however,

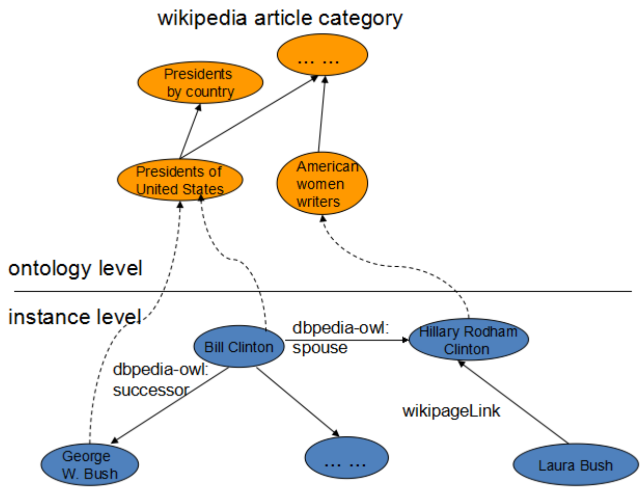

Figure 2: Referenced DBpedia Data

could also be applied to other knowledge base such as YAGO [8] or Freebase.

DBpedia is a crowd-sourced community effort to extract structured information from Wikipedia and make this information available on the Web. Every Wikipedia page has a corresponding concept in the DBpeida knowledge base. DBpedia is provided as a collection of files, or datasets, each with a specific type of data. Figure 2 shows the part of DBpedia data that we reference in this paper, that comprises instance level data and ontology level data. For the instance level data, we use the Wikipedia pagelinks dataset, that was created from the internal links between Wikipedia articles, and the mapping based infobox dataset, which is a clean version of the Wikipedia infoboxes, that together describe 4.58 million concepts and the relationship among them. We use the former dataset to measure context association between concepts, and the latter, which specifies relations like, e.g., relationship `dbpedia-owl:spouse` between concept `Bill Clinton` and concept `Hillary Rodham Clinton` – to measure structural association between concepts. For the ontology level data, we use the article categories dataset from DBpedia to provide the classification information for each instance and we use the taxonomy of categories dataset to measure the category association between concepts. The reason to use the article category dataset is that it has quite a large coverage and is constantly evolving.

4.2 Concept Graph Builder

Given a document, we first extract its concepts by employing a mention detection tool. The extracted concepts form the nodes of the concept graph. The edges of the graph represent the relationships between the concepts as derived from DBPedia, through the three kinds of association between concepts: context association, category association and structure association. In the following subsections, we describe how we measure the extent of these associations.

4.2.1 Context Association

Context association between two concepts reflects how often both concepts share contexts. In Wikipedia, we take shared contexts to be shared incoming links, i.e., pages that point to both concepts. A large number of shared incoming links indicates a high context association. Given two concept m_1 and m_2 we use the metric from [12] to measure their context associations. Let M_1 and M_2 denote the respective sets of incoming links to m_1 and m_2, then

$$ctxt(m_1, m_2) = 1 - \frac{\log(\max(|M_1|, |M_2|)) - \log(|M_1 \cap M_2|)}{\log|W| - \log(\min(|M_1|, |M_2|))} \quad (1)$$

where $|W|$ is the total number of concepts in the knowledge base.

4.2.2 Category Association

As mentioned in Section 4.1, DBpedia provides with category information for each concept, and a taxonomy of categories. Categories are intended to group together pages on similar subjects. Hence, if two concepts belong to the same Wikipedia category, they are associated with the same topic. Through the Wikipedia taxonomy of categories, We measure similarity of categories to evaluate the level of category association between concepts. Given two concepts m_1 and m_2, let $C_1 = \{c_{11}, c_{12}, \ldots, c_{1p}\}$, and $C_2 = \{c_{21}, c_{22}, \ldots, c_{2q}\}$ denote the respective sets of categories that m_1 and m_2 belong to. We measure the category association between these two concepts by first finding the pairwise similarity between any two individual categories c_{1i} and c_{2j}, and then combining the pairwise category similarities between all pairs of categories one from C_1 and the other from C_2, to form the groupwise similarity between C_1 and C_2, which is set to be the level of category association between the given two concepts. Next we define three workable pairwise similarity measures, and their combination into groupwise similarity.

Pairwise category similarity. Several works [13, 17] proposed to calculate the similarity of two categories in a taxonomy based on the specificity, or the information content of the categories. First, the information content score $IC(c)$ is computed for each node c in the taxonomy. Then given two nodes c_i and c_j in the taxonomy, let $MSCA(c_i, c_j)$ denote the common ancestor of c_i and c_j with highest information content, then the pairwise similarity of c_i and c_j is computed as follows.

$$pairwise(c_i, c_j) = \frac{IC(MSCA(c_i, c_j))}{IC(c_i) + IC(c_j)}$$

Different metrics are proposed to measure the information content of a node in the taxonomy [13]. The idea is that the more specific the node is in the taxonomy, the higher is its information content. In general, those metrics are either intrinsic or extrinsic. An intrinsic metric considers only the topological properties of the taxonomic backbone, whereas an extrinsic metric [13] also considers the instances that belong to each category in the taxonomy. In this paper, we try two intrinsic and one extrinsic metrics.

The first intrinsic metric uses the depth of the category in the taxonomy, i.e., its specificity, as its information content. Formally,

$$IC_{depth}(c) = \frac{\log(\max(depth(c)))}{\log(max\_depth)}$$

where max_depth denotes the maximum depth of the taxonomy. Note that the IC_{depth} assumes that the given taxonomy is a DAG (Directed Acyclic Graph). However, the taxonomy of Wikipedia categories does contain cycles. Moreover, the set of Wikipedia categories contains a large number of "Administrative" categories which define the status or stage of the pages, ignoring any semantic information about the page, and thus should not be considered in our metrics. In Appendix A we describe how we cleaned the taxonomy from cycles and administrative categories.

The second intrinsic metric considers the number of descendants of a category in the taxonomy to measure its information content. A large number of descendants indicates generality, or lower specificity. The descendants based information content of a category c is defined as

$$IC_{desc}(c) = 1 - \frac{\log(|D(c)|)}{\log(|C|)}$$

where $D(c)$ denotes the set of descendants of category c and C – the set of all categories in the taxonomy.

Extrinsic metrics consider also the number of instances that belong to a category. A large number of instances indicates a general, less specific category. For example, category `American` has a larger number of instances than category `Presidents of United States`. Formally,

$$IC_{inst}(c) = 1 - \frac{\log(I(D(c)))}{\log(I(C))}$$

where $I(D(c))$ denotes the number of instances that belong to category c or its descendants, and $I(C)$ denotes the total number of instances in DBPedia.

In Section 7, we compare these metrics experimentally.

Groupwise category similarity. Given two concepts m_1 and m_2 and their categories $C_1 = \{c_{11}, c_{12}, \ldots, c_{1p}\}$ and $C_2 = \{c_{21}, c_{22}, \ldots, c_{2q}\}$, for each $c_{1i} \in C_1$, find $best(c_{1i})$ which is the maximal pairwise similarity between c_{1i} and any category c_{2k} in C_2. Similarly, find $best(c_{2j})$ for each $c_{2j} \in C_2$. Groupwise similarity denoted by $cat(C_1, C_2)$) is defined as the average best similarity over all pairs,

$$cat(C_1, C_2) = 0.5 * \frac{\sum_{i=1}^{p} best(c_{1i})}{p} + 0.5 * \frac{\sum_{j=1}^{q} best(c_{2j})}{q} \quad (2)$$

4.2.3 Structure Association

Wikipedia infoboxes contain information about concepts and their various types of relationships, thus inducing a structural graph, $G(V, E)$, over the concepts, whose edges $e \in E$ are labeled by predicates $pred(e)$ that indicate the type of the relationship. We define the weight of an edge e to be

$$w(e) = \log(|pred(e)|)/\log(|E|)$$

where $|pred(e)|$ is the frequency of $pred(e)$ in E. Our intuition is that frequent predicates represent a general, less significant relationship. We thus define the structure association between two concepts m_1, and m_2, as the inverse of the sum of the weights of the edges $e_1, e_2, \ldots, e_k$ of the shortest path from m_1 to m_2:

$$struct(m_1, m_2) = \frac{1}{\sum_1^k w(e_k)} \quad (3)$$

To find the shortest path between m_1 and m_2, we use bi-directional breadth first search, and we set a threshold k to constraint the number of steps to guarantee performance. If we find no path within k steps, we set $struct(m_1, m_2) = 0$.

5. ASSIGNING WEIGHTS TO CONCEPTS

In the concept graph we generate for a given document, we assign weights to the concepts to reflect their relevance to the aspects of the document. We use two weight assignment methods: a global method that is based on the strength of the semantic relationships among the concepts, and a local method that is based on content similarity between the Wikipedia page of the concept and the given document.

5.1 Concept Graph based Weights

In graph theory and network analysis, centrality measurement is used to identify the most important nodes within a graph/network, the most influential persons in a social network, key infrastructure nodes in the Internet and more. According to different criteria of "importance" which suit different purposes, different graph properties of a node are considered for the evaluation of its centrality [3]. The most popular properties are: (1) degree, or number of ties that the node has, as high degree indicates popularity in the network, (2) closeness, or the inverse of the sum of the node's distances (length of shortest path) to all other nodes in the network, and (3) betweenness, or the number of shortest paths, between any two other nodes, that go through the node, as a node of high betweenness is critical to establish the short connections for other nodes in the network.

The set of concepts to represent the aspects of the document should be closely related, close to one another in the concept graph. We thus chose to measure centrality, or importance of a node, through its closeness property. Instead of using the shortest path between two concepts as in the traditional method, we define the distance between two nodes as the inverse of the weight of the edge between them in the concept graph. Let m_1 and m_2 be two concepts in the concept graph. We define

$$dist(m_1, m_2) = \frac{1}{\lambda_1 \cdot ctxt(m_1, m_2) + \lambda_2 \cdot cat(m_1, m_2) + \lambda_3 \cdot struct(m_1, m_2)} \tag{4}$$

where λ_i are wieght parameters of the three types of associations.

Subsequently, we define the closeness based centrality of a concept m as

$$centrality(m) = \frac{1}{|V|} * \sum_{m_j \in V} \frac{1}{dist(m, m_j)} \tag{5}$$

where V is the set of concepts in the concept graph.

5.2 Content based Weights

Complementing the global centrality measure for scoring a node in the concept graph is a local measure, being the similarity between the Wikipedia page that represents the concept and the document itself. We apply IR methods to measure this similarity, specifically, we use the simple tf-idf measure of document similarity. Formally, the content-based score of a concept m with respect to document d is defined as

$$content(m) = \frac{\vec{m} * \vec{d}}{\parallel \vec{m} \parallel * \parallel \vec{d} \parallel} \tag{6}$$

where $\vec{m}$ and $\vec{d}$ are the bag-of-words vector representations of m and d respectively.

We use the content-based weights of concepts for two purposes. First, we combine them with the centrality based weights. Second, we use them to filter out concepts that may be erroneously detected by the mention detection tool. Such errors may happen due to wrong disambiguation of concepts when mapping them to the knowledge base. A low content similarity of a concept is an indication that the concept is not related to the document. Therefore we remove concepts that have a content similarity lower than some threshold. For the remaining concepts we combine their centrality and content weights using a linear combination.

6. SEMANTIC RELATEDNESS OF DOCUMENTS

Given two documents, D_1 and D_2, we generate both respective concept graphs, and weight their nodes, as described in Sections 4 and 5, and turn to compute the semantic relatedness between these documents in terms of features of our concept graphs.

6.1 Pairwise Concept Similarity

A simple pairwise concept similarity is based on [12] which considers the shared incoming links in the knowledge base. We call it the Wikilink pairwise similarity, and use Eq. (1) to measure it.

We present another, novel pairwise similarity measure between two concepts that uses a neural network to represent concepts as continuous vectors.

6.1.1 *Concept2Vector representation*

We expand upon the Skip-Gram model by Mikolov et al. [11] to represent concepts as continuous vectors. We refer to our method as *Concept2Vector*. We first describe how we train the model and then how we use it for similarity.

Data Corpus preparation. To train a model for representing concepts as vectors, we need a data corpus that provides the context of each concept. We exploit the Wikipedia internal links for that. An internal link in a Wikipedia page (referred to as *wikilink*) comprises a reference (hyperlink) to another Wikipedia article and an optional surface form that represents the referenced article. Since each Wikipedia article corresponds to a concept in DBpedia, we consider each wikilink as a reference to a DBpedia concept. We preprocessed all wikipedia pages, replacing the surface form in each wikilink by the title of the referenced article. If the referenced article redirects to another page then we replace the surface form by the title of the target page. Consider, for example, the wikilink with a surface form `Oxford University` that references the page `University_of_Oxford`, and assume that the concept `Phi_Beta_Kappa` redirects to `Phi_Beta_Kappa_Society`. Then the text

"He is an alumnus of `Georgetown University`, where he was a member of `Kappa Kappa Psi` and `Phi Beta Kappa` and earned a `Rhodes Scholarship` to attend the `Oxford University`"

is replaced by

"He is an alumnus of `Georgetown_University`, where he was a member of `Kappa_Kappa_Psi` and `Phi_Beta_ Kappa_Society` and earned a `Rhodes_Scholarship` to attend the `University_of_Oxford`"

After pre-processing, each page in Wikipedia contains original words (that do not appear as part of a wikilink) and the concepts, each treated as one term.

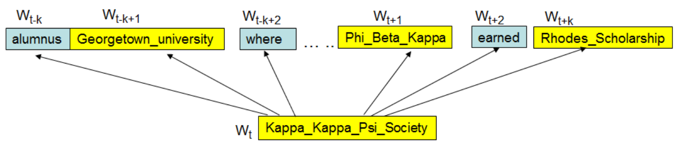

Figure 3: The Training Example for the Sentence

Model Usage. Based on Mikolov [11], Skip-Gram model is better for infrequent-terms representation. Since the occurrence of links in Wikipedia is sparse we choose the Skip-Gram model. The Skip-Gram model aims to maximize the probability of current term based on its surrounding terms. The parameters we used to train the model is as follows, window size = 10, sub-sampling = $1e-3$, cutoff min-count = 0. Figure 3 shows a model structure for the above text.

After the training, we obtain a vector for each term which could be either a word or a concept. All vectors have the same length, we took it to be 200. The model generates the vectors for 4.27 millions concepts in total. Compared with the total 4.58 millions concepts in DBpedia, the coverage of the concepts is 93.2%, which covers most of useful concepts already. The vector representation could be further improved by completing more Wikipeida links. It is observed that if a concept appears multiple times in a Wikipedia article, usually the Wikipedia link is only added for the first occurrence. Then we could try to add more links by using the exact matching in the same article. If a phrase in the following part of a Wikipedia article matches an explicit Wikipedia link, then we could add a new Wikipedia link on the phrase. In this way, more Wikipedia links could be generated, then we could obtain better and more complete context of a concept. Thus, the quality of the vectors could be improved.

The pairwise similarity between two concepts, m_1 and m_2, is defined as the cosine similarity between their respective Concept2Vector representations, $\vec{m_1}$ and $\vec{m_2}$:

$$Concept2VecSim(m_1, m_2) = \frac{\vec{m_1} * \vec{m_2}}{\|\vec{m_1}\| * \|\vec{m_2}\|} \quad (7)$$

We can also take the pairwise similarity to be a linear combination of the above, as described in the experiments section, Section 7.

6.2 Document Similarity

We extend the groupwise similarity measure of 4.2.2 by incorporating weights into the equation. Let $\{m_{1i}\}_{i=1}^{p}$ be the concepts in the concept graph of D_1 and $\{m_{2j}\}_{j=1}^{q}$ the concepts in the concept graph of D_2. Let $\{w_{1i}\}_{i=1}^{p}$ and $\{w_{2j}\}_{j=1}^{q}$ be their respective weights. We define the semantic relatedness between D_1 and D_2 as:

$$ConceptGraphSim(D_1, D_2) =$$
$$0.5 * \frac{\sum_1^p w_{1i} * best(m_{1i})}{\sum_1^p w_{1i}} + 0.5 * \frac{\sum_1^q w_{2j} * best(m_{2j})}{\sum_1^q w_{2j}} \quad (8)$$

where $best(m_{1i})$ ($best(m_{2j})$ respectively) is the best pairwise similarity for m_{1i} (m_{2j} respectively) over all concepts in $\{m_{2j}\}_{j=1}^{q}$ ($\{m_{1i}\}_{i=1}^{p}$ respectively).

To derive the final similarity score of two documents we combine the ConceptGraphSim with ESA [7] scores using a linear combination.

The above two similarity measures are quite different from each other. First, the ConceptGraphSim uses a small number of concepts detected directly from the document while ESA uses a large vector of concepts that are supposed to indirectly represent the document. Second, the ConceptGraphSim uses a unique aggregation of the pairwise similarities between pairs of concepts while ESA uses a cosine similarity between two concept vectors. It should be noted that the pairwise similarities of ConceptGraphSim between each pair of concepts uses a cosine-similarity between their continuous vector representations. Thus the ConceptGraphSim comprises both a pairwise similarity and cosine similarity.

7. PERFORMANCE STUDY

To verify the effectiveness of our method we conduct comprehensive experiments to verify the contribution of different parameters for the generation of the concept graph and for its application to document similarity. Our results show that the proposed concept graph captures the essential aspects of the document content and it outperforms other methods for semantic similarity of documents.

7.1 Experimental Testbed

Data Sets. We use the LP50 [10] dataset for evaluating our document semantic similarity. LP50 consists of 50 news articles from the Australian Broadcasting Corporation (ABC). The documents are of varying length, from 51 to 126 words and they cover a number of broad topics. Each pair of documents was judged by 8 to 12 different annotators, who rated the pairs on a scale of 1 to 5, where 1 indicates "highly unrelated" and 5 – "highly related". The final rate of each pair was taken as the average over all its judgments. The result is 1225 pairs with 67 distinct relatedness scores.

Evaluation metric. Similar to [10, 7, 16] we use the Pearson linear correlation to measure the agreement with the human judgments.

Experimental Setup. We make use of the TagMe [6] as our mention detection tool. As described in section 6.1.1 we processed Wikipedia to generate the Concept2Vector model. We generated vectors for 3 million words and for 4 million concepts. The vector for each term has 200 dimensions. We set the threshold for filtering our concepts with a low content similarity (section 5.2) to 0.001. We test a variety of configurations of our method. The various parameters and their values are shown in Table 1 with a reference in brackets to the section where they are described. To evaluate each parameter separately we fix the other parameters and vary that parameter. The values in bold are the default values that we fix when evaluating a specific parameter.

As we show in the rest of this section, the best results were achieved using a simple linear combination of all described parameters. Specifically for assigning weights to edges in the concept graph of a document, the best results were achieved when using a linear combination of ctxt (Eq. 1), cat (Eq. 2) and struct (Eq. 3) similarity between two concepts. For assigning weights to concepts in the concept graph, the best result was achieved when using a linear combination of the centrality based weights (Eq. 5) and content based weights (Eq. 6). For the pairwise similarity between a pair of concepts, the best result was obtained by a linear combination of Concept2VecSim (i.e the Cosine similarity between the vector representations of the two concepts) and the ctxt based

Table 1: Configuration Parameters

Parameter	Symbol	Value
Category association (4.2.2)	IC	**depth**, desc, inst
Concept Graph Centrality(5.1)	Dist	ctxt, cat, struct, **ctxt+cat+struct**
Concept weights (5)	Weight	none, centrality, content, **centrality + content**
Pairwise Concept similarity (6.1)	conceptSim	ctxt, Concept2VecSim, **ctxt + Concept2VecSim**
Document similarity (6.2)	documentSim	**ConceptGraphSim**, ESA, ConceptGraphSim + ESA

similarity (Eq. 1). Finally we show that for the overall document similarity, a linear combination of our ConceptGraphSim method with ESA yields the best results. We leave it for future work to learn the parameters automatically.

All the experiments were conducted on a 2.8GHz Intel Core machine with 8GB main memory running Windows 7, and all our algorithms are implemented in Java.

7.2 Compare to state-of-the-art

We compare the best configuration of our method (using the bold values of each parameter in Table 1) to the following state-of-art methods in document similarity measurement, i.e. Latent Semantic Analysis (LSA) [5], Explicit Semantic Analysis (ESA) [7], graph model (GED) [16], WikiWalk [19], and ConceptLearned [9]. Table 2 shows the results. Our method combined with ESA, termed `ConceptGraphSim + ESA` achieved 0.786 Pearson correlation which outperformed all other methods except ConceptLearned [9] which achieved 0.808. We should note that ConceptLearned [9] uses a relatively large number of 17 features compared with our system. Moreover, their weights are learned and tested using k-fold cross validation on the same dataset, thus it might be overfitted to the LP50 benchmark. Our method is not domain specific. Moreover, we believe that we could further improve our performance on that dataset by combining more features and learning their weights, but we leave it for future work.

Among the other methods, ESA achieved 0.72 Pearson correlation. The method `ESA Implemented` is our implementation of ESA where we index the Wikipedia documents into a Lucene[1] index. To generate the ESA concepts vector for a given document, we use the document content as a query to the Lucene index with a BM25 similarity model. Similar to [19] we take the top-625 results as the ESA concepts. We got some improvement 0.727 when we add to the generated query, pairs of terms from the document content in a sliding window of size 7. Our method without ESA, termed `ConceptGraphSim` achieves 0.745 correlation (p-value $= 2.02E - 217$).

In general, both our method and ESA use a set of weighted concepts to represent the document. Yet, there are several differences between the two. First, ESA uses a large number of concepts while we use only the concepts that are detected in the document. On average we use about 20 concepts for each document while ESA uses 625 concepts (lower numbers achieved worse performance for ESA). Second, ESA uses just a cosine similarity between two vectors while we explore all pairwise similarity between pairs of concepts and take the best matching combination.

[1]http://lucene.apache.org/

Table 2: Comparison with Existing Methods on the LP50 dataset

Parameter	Pearson correlation
LSA [5]	0.60
GED [16]	0.63
ESA paper [7]	0.720
ESA implemented	0.727
ConceptGraphSim	0.745
WikiWalk + ESA [19]	0.772
ConceptGraphSim + ESA	0.786
ConceptsLearned [9]	0.808

Our hypothesis is that the two methods (ConceptGraphSim and ESA) complement each other since each uses a different set of concepts and a different similarity measure. We tried therefore to combine the two methods using a linear combination. This follows the work of WikiWalk [19] that also builds on top of ESA. They use personalized PageRank on all Wikipedia pages where the personalized teleport vector uses the ESA weights on concepts detected in the document. As can be seen in Table 2, our method combined with ESA achieve a higher correlation than WikiWalk with ESA. It should be noted that WikiWalk report that their ESA implementation achieved correlation of 0.766 so their improvement over ESA is quite negligible. Moreover, the WikiWalk implementation for document similarity requires to run a personalized PageRank for each document which is quite inefficient compared with our derivation of weights.

Next we report the parameter tuning of the different configurations of our method. Please be noticed that the performance is reported without the combination with ESA.

7.3 Effect of Concept Weight Strategies

As discussed in Section 5, we provide two strategies to assign weights to the concepts in the concept graph. The first strategy, termed `centrality`, is based on the closeness centrality of concepts in the concept graph (Eq. 4). The second strategy, termed `content` is based on cosine similarity between the bag-of-words vectors of the document and the Wikipedia page that represents the concept (Eq. 6). Furthermore, for the centrality based weights we presented three different methods (termed ctxt, cat and struct) to measure the similarity between pair of concepts. Table 3 shows the effect of the different weighting strategies on the final results when fixing all other parameters to their default values.

For the centrality weights (Eq. 4) we try some simple linear combinations of the three association methods, namely ctxt (Eq. 1), cat (Eq. 2) and struct (Eq. 3) . We try using the ctxt level only (setting $\lambda_1 = 1$, $\lambda_2 = 0$ and $\lambda_3 = 0$), combining the ctxt with struct (setting $\lambda_1 = 1$, $\lambda_2 = 0$ and $\lambda_3 = 1$) and combining the three kinds of associations together. We can see that for the centrality weights, the best correlation reaches 0.683 when using the three association methods together. We tried also several weighted combination for the centrality weights with different values of the $\lambda_1, \lambda_2, \lambda_3$ but we got similar results.

Table 3: Effect of Centrality Metrics and Concept Weight Strategies

Weight,Dist		**Pearson Correlation**
Concept Association for	ctxt	0.663
	ctxt+struct	0.672
	ctxt+cat+struct	0.683
Centrality Concept Weights	none	0.655
	centrality	0.683
	content	0.736
	centrality + content	0.745

Table 4: Effect of Pairwise Concept Similarity

conceptSim	**Pearson Correlation**
ctxt	0.725
Concept2VecSim	0.732
ctxt+Concept2VecSim	0.745

For the concept weights parameter, we see that the baseline method which uses uniform weights (line that corresponds to "none" in the table) got correlation of only 0.655. The content-based weights achieve a correlation of 0.736 which is better than the correlation of 0.683 by the centrality-based weights. It should be mentioned that the content-based weights are similar to the weights used in ESA. Still we show that by combining the centrality weights derived from the concept graph with the content-based weights we get an improved correlation of 0.745 compared to each method separately.

7.4 Effect of Pairwise Concept Similarity

Table 4 shows the results for varying the Pairwise concept similarity scorers while fixing all other parameters to their default values. Our proposed Concept2VecSim method which exploits the neural network representation of concepts (Eq. 7) achieves correlation of 0.732 which is better than the wikilinks based similarity, termed `ctxt`, (Eq. 1). In addition, the Concept2VecSim is more efficient than the wikilinks based similarity since it uses a cosine-similarity between the continuous vectors of the two concepts which are of size 200. In comparison, the ctxt similarity intersects two sets of incoming links which could have a larger number of elements.

Similar to the phenomena presented in the previous section, the combination of several methods achieves better results than each method separately. We can see that using a simple linear combination of ctxt with Concept2VecSim method yields a correlation of 0.745 which is better than the correlation of the two similarity measures separately.

7.5 Effect of IC Metrics for Category level Association

Table 5 summarizes results of experiments with different IC (Information Content) metrics for calculating category level association, as discussed in Section 4.2. We can see that the three methods achieve quite similar results.

8. CONCLUSIONS

We presented `concept graph` for representing a document using its detected concepts. The concept graph utilizes DBPedia to explore relationships among the detected concepts and weighs the concepts according to their closeness centrality to reflect their relevance to the aspects of the document. We then presented a novel similarity measure **`ConceptGraphSim`** between two documents by comparing their concept graphs. The similarity measure first represents concepts as continuous vectors using neural network and then combines pairwise similarity between pairs of concepts using their continuous vector representations and their assigned weights. Our ConceptGraphSim combined with ESA achieved Pearson correlation of 0.786 on the LP50 dataset which outperformed other methods, except one system [9] that achieved a better Pearson correlation of 0.808. We believe that by adding more features as they did we can improve our results. We further believe that the combination of concept based centrality with neural networks as we demonstrated in our system, has a high potential that can be further exploited to achieve better results.

We conducted an extensive evaluation on the various parameters of our method. We show that by combining several weighting methods and several pairwise concept similarity measures we improve over each method separately. For example we show that the best strategy to assign weights to concepts is achieved by combining their centrality-based weights with content-based weights. Similarly, we show that combining the pairwise similarity between continuous vector representations of concepts with a pairwise similarity that considers associated context, improves over each method separately.

For future work, we plan to add more features and to improve the quality of the concept graph representation by either improving the accuracy of the mention detection tool or enriching the concept graph with additional related concepts from the knowledge base.

Table 5: Effect of IC on Category Level Association

Parameter	**Pearson Correlation**
depth	0.745
desc	0.745
inst	0.746

9. REFERENCES

[1] C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, and S. Hellmann. Dbpedia – a crystallization point for the web of data. *Web Semantics*, 2009.

[2] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research (JMLR)*, 2003.

[3] S. P. Borgatti. Centrality and network flow. *Social Networks 27*, 2005.

[4] J. Daiber, M. Jakob, C. Hokamp, and P. N. Mendes. Improving efficiency and accuracy in multilingual entity extraction. In *Proceedings of the 9th International Conference on Semantic Systems (I-Semantics)*, 2013.

[5] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 1990.

[6] P. Ferragina and U. Scaiella. Fast and accurate annotation of short texts with wikipedia pages. *IEEE Software 29(1)*, 2012.

[7] E. Gabrilovich and S. Markovitch. Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, 2007.

[8] J. Hoffart, F. M. Suchanek, K. Berberich, and G. Weikum. Yago2: A spatially and temporally enhanced knowledge base from wikipedia. *Artificial Intelligence*, 2013.

[9] L. Huang, D. Milne, E. Frank, and I. H. Witten. Learning a concept-based document similarity measure. *Journal of the Association for Information Science and Technology*, 2012.

[10] M. D. Lee, B. Pincombe, and M. Welsh. An empirical evaluation of models of text document similarity. In *Proceedings of Annual Conference of the Cognitive Science Society (CogSci)*, 2005.

[11] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. In *Proceeding of the International Conference on Learning Representation (ICLR) Workshop*, 2013.

[12] D. Milne and I. H. Witten. An effective, low cost measure of semantic relatedness obtained from wikipedia links. In *Proceedings of AAAI Workshop on Wikipedia and Artificial Intelligence*, 2008.

[13] P. Resnik. Using information content to evaluate semantic similarity in a taxonomy. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, 1995.

[14] G. Salton. *The SMART Retrieval System: Experiments in Automatic Document Processing.* Prentice - Hall, Upper Saddle River, NJ, USA, 1971.

[15] G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Commun. ACM*, 1975.

[16] M. Schuhmacher and S. P. Ponzetto. Knowledge-based graph document modeling. In *Proceedings of the ACM International Conference on Web Search and Data Mining (WSDM)*, 2014.

[17] N. Seco, T. Veale, and J. Hayes. An intrinsic information content metric for semantic similarity in wordnet. In *Proceedings of 16th European Conference on Artificial Intelligence*, 2004.

[18] P. D. Turney and P. Pantel. From frequency to meaning: Vector space models of semantics. *Artificial Intelligence Research*, 2010.

[19] E. Yeh, D. Ramage, C. D. Manning, E. Agirre, and A. Soroa. Wikiwalk: Random walks on wikipedia for semantic relatedness. In *Proceedings of the 2009 Workshop on Graph-based Methods for Natural Language Processing*, 2009.

APPENDIX

A. CLEANING WIKIPEDIA TAXONOMY

In Section 4.2.2, we mentioned that the computation of similarity between Wikipedia categories through the taxonomy of categories is based on the assumption that the taxonomy is acyclic. The taxonomy is defined in the Wikipedia category dataset of DBpedia by the skos:broader relationship specified between category pages. The taxonomy is the implied directed graph, whose nodes are the categories, and a directed edge leads from node c_1 to node c_2 for each pair of categories c_1 and c_2 such that c_2 is specified as skos:broader in the page of c_1. Namely, an edge leads from a category to a broader, more general category. That directed graph is supposed to be acyclic, inducing a global hierarchical taxonomy of categories. However, as Wikipedia categories are defined and assigned in the hierarchy by humans, cycles are unavoidable. For example, the cycle shown in Figure 4 is implied by the skos:broader relations found in DBpedia pages of categories.

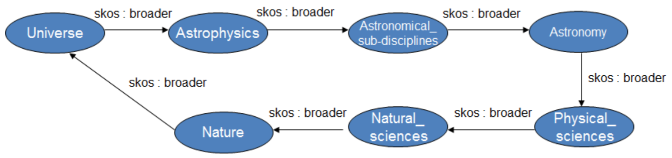

Figure 4: A Cycle in Wikipedia Category Hierarchy

We thus conducted a preprocessing step to remove edges from the category hierarchy to make it acyclic. Detecting a cycle, we removed an edge from it, which was selected by the following heuristic. For each node v in the cycle, we count the number, $desc(v)$, of its descendants in the taxonomy, which are the nodes from each of which a directed path leads to v, a path which does not visit any node on the cycle but v. Now, since every edge $e = (s_e, t_e)$ leads from a category s_e to a broader category t_e, we expect $desc(t_e)$ to exceed $desc(s_e)$. Our heuristic is thus to remove an edge $e = (s_e, t_e)$ where $desc(t_e) < desc(s_e)$. If multiple edges satisfy this requirement, we remove the one with largest difference $desc(s_e) - desc(t_e)$. Based on this heuristic, we have removed 2523 edges in total, leaving 1824523 edges in place.

Another issue mentioned in Section 4.2.2 is that the set of Wikipedia categories also contains the administrative categories which are intended for use by editors or by automated tools, based on features of the current state of articles. For example, the concept Albert Einstein belongs to the category Articles with unsourced statements. These administrative categories hinder the evaluation of category level association, since two concepts that belong to the same administrative category do not have semantic similarity. There is no explicit indication of an administrative category, and we tried to identify them through their being subcategories of $Category : Wikipedia\_administration$. However, we could not simply remove a whole subgraph, since it may include many topic bearing categories. Wikipedia maintains some naming conventions for editors to assign the title on each Wikipedia page. Relying on these, we propose the following three conditions on the category name, aka label, each of which implies an administrative categories. i) The label begins with WikiProjects, Wikipedia, or Articles, e.g., `Category:Articles_needing _translation_from_French_Wikipedia` ii) The label ends with stubs, templates, e.g., `Category:England_RL_templates` iii) The label contains missing, cleanup, unknown, pages, e.g., `Category:Year_of_death_missing`

Employing the above conditions, we identified and removed 62509 administrative categories, leaving 798962 categories in place.