Towards Semantic Quality Enhancement of User Generated Content

José María González Pinto[1], Niklas Kiehne[2] and Wolf-Tilo Balke[1]

{pinto,balke}@ifis.cs.tu-bs.de[1] n.kiehne@tu-bs.de[2]

Institut für Informationssysteme, TU Braunschweig
Mühlenpfordstrasse 23, 28106 Braunschweig Germany

Abstract. With the increasing amount of user-generated content such as scientific blogs, questioning-answering archives (Quora or Stack Overflow), and Wikipedia, the challenge to evaluate *quality* naturally arises. Previous work has shown the potential to evaluate automatically such content focusing on syntactic and pragmatic levels such as conciseness, organization, and readability. We push forward these efforts and focus on how to develop an intelligent service to ease the engagement of users in two *semantic attributes*: factual accuracy, e.g., whether facts are correct and validity, e.g., whether reliable sources support the content. To do so, we deploy a Deep Learning approach to learn citation categories from Wikipedia. Thus, we introduce an automatic mechanism that can accurately determine what *specific citation category* is needed to help users increase the value of their contribution at a semantic level. To that end, we automatically learn *linguistic patterns* from Wikipedia to support a broad range of fields. We extensively evaluated several machine learning models to learn from more than one million annotated sentences from the massive effort of Wikipedia contributors. We evaluate the performance of the different methods and present a profound analysis focusing on the balance accuracy achieved.

Keywords: automatic quality enhancement, user-generated content, data curation.

1 Introduction

With the continuously growing of user-generated content on the Web, the challenge to verify quality content naturally arises. Previous work has shown that the quality of user-generated content is a combination of independent assessments from different indicators [13]. For instance, consider an article and the following indicators 'readability', 'factual accuracy' and 'validity.' Readability corresponds to a dimension called *syntactic* and is rather easy to assess because it is concerned with the structure of the content. In contrast, factual accuracy corresponds to the dimension *semantic* and is more difficult to assess, e.g., how do we know if the statements in the content in the article are correct? What about validity, e.g., whether reliable sources support the content? We argue that both of these issues could be tackled with proper citations, but how

can a user decide if a statement - a claim - requires a citation and of which type: a book, a scientific paper, a journal paper? To answer these questions, we should look at Wikipedia's remarkable example. Wikipedia is the most successful collaborative effort to create encyclopedic content freely available on the Web. Its popularity as a source of information as measured by Alexa[1] makes Wikipedia the top 5 ranked website in the world as stated in November 2017. Researchers assessed Wikipedia's quality for specific domains, and surprisingly it has been found to achieve comparable scores as expert-curated encyclopedias like, e.g., Encyclopedia Britannica [21]. On the other hand, in some cases, the completeness of the content of Wikipedia has been shown to be one of its drawbacks for more specific domains [5, 6, 16, 24]. In any case, as stated by Stvilia et al. [25] *"What makes special Wikipedia as a resource is that the quality discussions and processes are strongly connected to the data itself [...]".*

Thus, even though no final agreement exists on what definitely can be 'the metric' that can measure the quality of Wikipedia, citations play a key role, i.e., valid citations provide *credibility* to the Wikipedia's content. As the Wikipedia 'citing sources' page[2] states: "*Wikipedia's verifiability policy requires inline citations for any material challenged or likely to be challenged and for all quotations, anywhere in article space".* Moreover, in the Wikipedia's verifiability page[3] we find that *"[...] verifiability means that other people using the encyclopedia can check that the information comes from a reliable source. Verifiability, no original research and neutral point of view are Wikipedia's core content policies".*

To motivate collaboration - and quality - on the content, Wikipedia uses the tag 'citation needed.' This tag is a request for other editors to verify a statement by a proper citation. Consider the following example[4]: *"Deep learning architectures are often constructed with a greedy layer-by-layer method* [citation needed]*."* Among the citations categories that exist on Wikipedia, what category would be needed here? A book, a journal, a conference paper? Is it possible to learn *citation categories accurately* from the linguistic patterns that exist on Wikipedia? If we could do it, then we could help users to enhance their content at the *semantic* level by tagging statements in need of a specific citation category! We argue that two of the semantic features defined in Dalip et al. [13] could be tackled: *factual accuracy,* e.g., whether facts are correct and *validity,* e.g., whether reliable sources support the content.

To this end, we focus on this paper in assessing the current state of the art machine learning approaches including classical techniques and deep leaning techniques exploiting different embedding word representations to learn citation categories automatically from Wikipedia. Our contributions are as follows:

- We introduce a novel automatic method with the potential to enhance user-generated content quality inspired by Wikipedia's successful qualitative practices.
- A comparative study of competitive methods to answer a simple but powerful question: what category of citation is needed?

[1] https://www.alexa.com/siteinfo/wikipedia.org
[2] https://en.wikipedia.org/wiki/Wikipedia:Citing_sources
[3] https://en.wikipedia.org/wiki/Wikipedia:Verifiability
[4] https://en.wikipedia.org/wiki/Deep_learning

- A detailed analysis and discussion that provide insights of what the best model can achieve and what this can tell us about the feasibility of an Intelligent Open Service available for the community to use.

2 Related Work

Our work draws motivation from recent qualitative studies on Wikipedia. Specifically, research efforts that use machine-based approaches to assess the quality of the massive effort behind Wikipedia. We will first start with a general discussion on information quality and then on the specific efforts related to our work: machine-based approaches to quantify the quality of Wikipedia.

There is a body of research on various aspects of information quality in information management sciences [10, 18, 19]. Quality is considered a multi-dimensional concept that involves different criteria such as accessibility, accuracy, authority, see [25] for a comprehensive discussion on the topic related to Wikipedia.

Dalip et al. [13] provided a Multiview framework for assessing the quality of collaboratively created content. As the authors stated in their extensive analysis of indicators "…it is clear that most of the effort to assess quality in collaborative repositories has focused on the syntactic and pragmatic levels. Indicators for these levels are normally easy to calculate, in contrast to semantic indicators that often require the use of expensive natural language processing techniques". Indeed, this is confirmed in several previous works to assess the content quality of Wikipedia automatically. Most of them use simple metrics such as article length, word count, sentence count, interlink count, and edition history. Herein we present some representative efforts that account for some of these features. For instance, the work of Adler and Alfaro [1] introduces a system to measure authors reputation of Wikipedia based on content. To measure an author's reputation, the researchers rely on the idea of "content evolution"; the assumption behind this approach is that content preserved should have more credibility. Concretely, authors lose reputation when their edits are rolled back or undone. This reputation mechanism aimed at introducing the idea of "flagging" new content as coming from an author with high-reputation or low-reputation. Such mechanism could in theory benefit jointly efforts of improving Wikipedia content by other volunteers to focus on content that comes from authors with low-reputation. Quant et al. [7] explored deep learning techniques for content-based analysis for classifying articles using the content assessment currently performed by Wikipedia's editors. The models they used did not consider citations or any other metadata for the task.

3 Approach and Problem Formulation

3.1 Problem Formulation

In this section, we define the problem and provide definitions to accomplish our goal: a machine-based approach to learn from linguistic patterns, Wikipedia citation categories.

Definition 1. Citation is a reference to an external source used to support what is mentioned in the text –usually a sentence- in a Wikipedia article. In Wikipedia there exist the following 16 categories of citations: web, news, book, journal, encyclopedia, episode, map, press release, video game, comic, conference, av. Media, court, thesis, mailing list, and newsgroup.
Definition 2. Claim: a claim in this work is a sentence in Wikipedia that needs or has a specific citation category.
Let k be the number of citation categories in Wikipedia. Our task is to find for a given claim c its corresponding citation category.
Claim Citation Categorization Problem. Formally, we want to learn a function: $f: \mathbb{R}^n \rightarrow \{1, \dots, k\}$. When $y = f(x)$, the model assigns an input described by vector x to a category identified by numeric code y. The vector x in our case is a vector representation of a claim c. Different alternatives exist to learn from data such a function and we will explore some of them in this work.

3.2 Models

In this section, we describe the models we evaluated solve our problem. Firstly, we will start with the sequence-based models (LSTM and BiLSTM); secondly, we present non-recursive methods: Convolutional Neural Networks and Fully Connected Networks; and thirdly, we present other more "classical" machine learning approaches used as baselines. The model's architecture and source code used in the paper are available on request.
LSTM. One of the most widely successfully used recurrent neural networks is the Long-Short-Term-Memory (LSTM) by Hochreiter and Schmidhuber[14]. Researchers have applied LSTM to several sequence data problems, such as sentence embeddings for information retrieval [23], speech recognition from audio data [11] or the translation of sentences into different languages [2]. See the work of [12] for a recent extensive empirical study of different variations of LSTM's performance and tuning of its parameters.
Bidirectional LSTM. The bidirectional LSTM architecture is very similar to the unidirectional architecture. The only notable difference is how the input data is presented. In a common recurrent network, the data is processed from the beginning of the sequence to the end of it. Therefore, it is impossible for the model to know anything about the following sequence element before reaching it. In the case of natural language sentences, this restriction of context poses a problem since the semantics of a word usually depend on past and future context as well. To overcome this limitation of unidirectional recurrent networks a bidirectional layer is used. The idea is to add another LSTM parallel to an existing one, but with a reversed input. Both LSTMs work on their version of the input data, but their output is concatenated, such that any following layer will have access to both past and future context. Naturally, this doubles the number of weights and also the computation time needed, making this architecture even slower than a usual LSTM. The architecture used here consists of two bidirectional LSTM

layers followed by three dense layers with a dropout of 25%. This model is by far the biggest architecture regarding weights.

Convolutional Neural Network. Convolutional neural networks (CNN) are known best for their use on image data, such as object detection [17] or even face recognition [26]. Recently, they have also shown to achieve state of the art results in sentence categorization. For instance, Kim [15] proposed not only to look at one specific amount of word vectors but also to use multiple context sizes in parallel. The idea here is as Zhang, and Wallace [27] demonstrated: to capitalize on the distributed representation of words embeddings. Zhang and Wallace provided practical guidelines of what can be achieved using CNN for text classification tasks. The implementation used here consists of four different context windows of two, three, four and five nearby words. Goodfellow, I. et al. [9] have emphasized that three essential ideas motivate the use of CNN's in different machine learning tasks, including text classification: sparse interactions, parameter sharing, and equivariant representations. For text classifications task, sparse interactions allow for learning linguistic n-grams patterns; parameter sharing influences computation storage requirements, and equivariant representation allow for robustness in the patterns learned regarding of the position in the sentence.

Feed Forward Nets. These fully connected neural networks represent one of the most straightforward architectures available. The model uses fixed length input vectors from layer to layer sequentially, and with no recursion. These models are usually not able to compete with the more complex approaches for the sequence data, such as LSTMs. On the other hand, the simpler the model, the shorter is the training and inference time.

Other classical machine learning approaches. For the sake of completeness, we applied other well-known "classical" Machine Learning algorithms to the fixed size data set. AdaBoost, Random Forests, and Decision Trees were chosen from the Scikit-learn[5] Python library to provide alternatives for performance comparison.

4 Experimental Setup

We work on the Wikipedia dump introduced by Fetahu, B. et al. in [8]. The data format consists of a big CSV file with six columns and a total of 8,797,947 rows of samples. In Table 1 we show the structure of this big table in CSV form.

Table 1. Raw Data Format

Column	Explanation
Entity	The Wikipedia article the current sample was taken from.
Section	The name of the section the sentence was taken from.
Sentence	The actual sentence encoded in Wikipedia markup language.
Citation	Wikipedia markup describing the type of citation.
URL	A hyperlink to the source that was referenced.
Cite Type	The category of citation that was used.

[5] http://scikit-learn.org/stable/

4.1 Data Preprocessing

Sentence Representation. To use the models described in the previous section, we transformed the data into vectors or sequences of vectors. Since the original base data comprises an extensive vocabulary of over 700k unique words (with capitalization) typical approaches like Bag of Words would result in massive vectors.
Because a sentence usually consists of less than 100 unique words which cause the Bag of Words matrix to be extremely sparse. Instead of extremely sparse representation, we opted for the use of dense word vectors, also called word embeddings. In particular, for the word2vec algorithm by Mikolov et al. [22]. In a nutshell, word embeddings pack more information into far fewer dimensions. Researchers have shown [27] two approaches to take advantage of word embeddings for classification tasks: 1) learn word embeddings jointly with the problem at hand and 2) use embedding vectors from a precomputed embedding space that might exhibit useful properties (captures general aspects of language structure). In this work, we push this idea further by comparing precomputed vectors of word2vec (trained on Google News dataset) and FastText [4] (trained on Wikipedia) with our learned word2vec vectors. FastText is another neural language model that accounts for more syntactic properties in the language and deals with out of vocabulary words by learning representations of words as the sum of character n-grams vectors [4].

In the following, we describe all the necessary preprocessing steps to obtain sequences of word vectors. The steps are given in the order of their execution unless stated otherwise.
Duplicates removal. We observed that there are sentences in the data that have more than one citation, even more than one citation category. Because in this work we restrict the problem to assign one category to each sentence, we used only sentences that have precisely one citation and are therefore suitable representatives of their respective classes. We leave the problem of assigning more than one category for future work. Thus, with this preprocessing, the corpus shrinks to 5,695,580 samples and 17 distinct citation categories as shown in Figure 1.

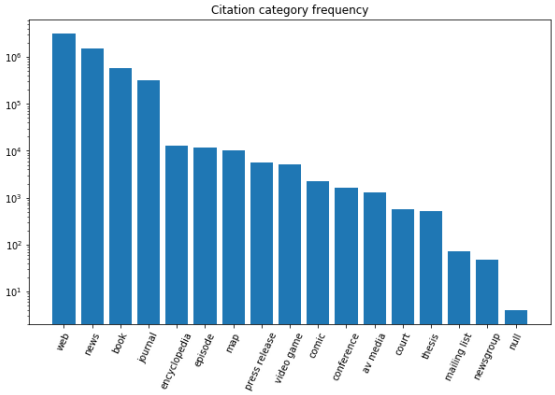

Figure 1. Wikipedia Citation category frequencies.

Parsing and tokenization. At this point, the sentences are still encoded in Wikipedia markup and therefore not immediately usable. To extract the natural language parts, we used the MediaWiki Parser[6]. After the extraction of the sentences, we tokenize them using the NLTK library. More specifically, the NLTK TreeBankWordTokenizer implementation and a PunktSentenceTokenizer [3]. The latter is needed to split paragraphs into multiple sentences which the word tokenizer then further splits down into tokens. Our data now consists of 5,695,580 samples, with a total of 11,121,707 sentences and 264,943,821 tokens.

Punctuation removal. After tokenization, we filter the punctuation out. Thus, the number of tokens reduces to 231,857,810.

Word2Vec representation. A crucial part of the data preparation is the computation of word vector representations. The training is carried out by the Gensim Python[7] library which includes a fast and efficient implementation of the Word2Vec algorithm optimized for multiple CPUs. There is a multitude of parameters that strongly impact the quality of the vector representations, e.g., the dimensionality of the word vectors, the size of the context window, the learning rate, the numbers of training epochs or the number of negative samples. However, how can the quality of word vector representations be measured? Mikolov proposed a test based on analogical reasoning with which it is possible to test for semantic and syntactic analogies within the model. Such a test set contains multiple predefined relationships such as "Berlin is to Germany as Madrid is to Spain". If the model successfully learned the syntactic relationship between countries and their capitals, then the difference vector of Germany and Berlin should be very similar to the difference vector of Madrid and Spain.

Table 2. Comparison of word vector model performances

Category	**Word2Vec Google**	**Word2Vec Wikipedia**	**FastText Wikipedia**
capital-common-countries	0.8357	0.9571	0.9625
capital-world	0.8490	0.9431	0.9458
currency	0.3889	0.0000	0.0143
city-in-state	0.7558	0.6096	0.7364
family	0.9477	0.8708	0.8889
gram1-adjective-to-adverb	0.3730	0.2900	0.5517
gram2-opposite	0.4917	0.3889	0.6128
gram3-comparative	0.9143	0.8350	0.8534
gram4-superlative	0.8553	0.8476	0.8203
gram5-present-participle	0.8241	0.8034	0.7137
gram6-nationality-adjective	0.9679	0.9707	0.9869
gram7-past-tense	0.6776	0.7396	0.6684
gram8-plural	0.8067	0.7685	0.8908
gram9-plural-verbs	0.7138	0.6571	0.6779
Total	0.7728	0.7766	0.8104

[6] https://github.com/earwig/mwparserfromhell

[7] https://radimrehurek.com/gensim/

Google released a collection of such analogies and are available on the author's website[8]. It contains 20k different examples that help to discover the strengths and weaknesses of any Word2Vec model. Google also supplied a model trained on approximately 100 billion words to use as a baseline. In Table 2, we show a comparison of the model trained on the Wikipedia sentences to Google's pre-trained model and a FastText' pre-trained model on Wikipedia. The result indicates that the model on Wikipedia has learned analogies and thus, is ready to use for word vector representation.

Word2Vec parameters. The parameters that performed best are a vector dimensionality of 300, a context window of 10 words and 15 negative samples. The learning rate started at 0.05 and was reduced by ten percent after each of the 20 epochs. Additionally, we discarded words that occurred less than five times in the corpus because there is not enough data to learn good vector representations for them.

Data cleansing. Once the Word2Vec model is available for translating sequences of words into sequences of vectors, further cleaning of the data is required. To further improve the quality of the data, we discarded all samples that consist of more than one sentence (after tokenization). Focusing on single sentence samples lowers the number of total samples to 3,098,423.

Excluding infrequent categories. We also observed that some citation categories have a small number of samples. Consider for instance the categories mailing list, newsgroup with 33 and 17 samples respectively. In contrast, the category "journal" appears more than 400K times. Following the suggested best practices of [9], we exclude categories with less than 5000 samples. Therefore, we reduce the total corpus size to 2,470,703 samples.

Citation category inconsistency. There is another quality aspect hidden inside the data. As shown in Figure 1, the citation category "web" is the most frequent label, followed by "news", "book" and "journal". However, users wrongly used category "web" in some of the citations that originate from a newspaper, a book or a journal. One reason why this phenomenon happens is that all of these sources might also be available online. However, how to decide whether a citation belongs to the web category or not? Instead of solving this question the problem is dodged by splitting the corpus into two datasets. The first dataset is the corpus so far, but the second set excludes every sample labeled as web-content. The idea is simple: if the models perform significantly better on the second dataset, then it will confirm that the web category needs much cleaning to be of any use.

Sequence Representation. Many Machine Learning algorithms cannot operate on sequence data, whether the sequences have the same length or not. One solution is to summarize the sequences to map them to a fixed length vector. After preliminary experiments, we chose 30 as the fixed length. The mappings investigated here are the sum, mean or weighted sum of all the word vectors in a sentence. The latter operation multiplies each word vector by its corresponding TF-IDF[20] score before taking the sum which, in theory, should help to better represent a sentence as a sum of its words. The idea is to give more importance to those words that determine the content. Another important aspect is that these methods drastically reduce the memory consumption

[8] https://github.com/nicholas-leonard/word2vec/blob/master/questions-words.txt

since every sequence of 30-word vectors is represented as a single vector. Finally, in Table 3 we show an overview of the main characteristics of the two datasets used for our experiments.

4.2 Results

We used different implementations of the algorithms to support a more diverse set of performance comparison using the two datasets. Each dataset comes in four variants: sequences, summed, averaged or weighted summed sequences. The shape of the raw sequence data is thirty-word vectors per sample whereas the shape of the summarized view is just one-word vector per sample.

Metrics. We applied two different metrics to measure the effectivity of the proposed architectures. The accuracy metric measures the percentage of right decisions a classifier made. However, the accuracy does not include the class frequencies. For example, in a two-class problem, if one of the classes occurs in 99% of the cases, the easiest way to achieve an accuracy of 0.99 would be to classify everything as the most frequent class. Naturally, such a classifier would not have learned any useful rules to distinguish the classes.

To measure whether a model has learned to differentiate the classes or not, we used balance-accuracy. The idea is to compute the mean per-class accuracy. So, the classifier in the example above would score the per class accuracies of 0.99 and zero, resulting in a balanced accuracy of 0.495. A classifier is considered right as soon as both metrics are comparatively high and do not differ by a high margin.

We show in Table 4 and Table 5, the performances of the architectures outlined in previous sections. The Feed Forward models are abbreviated, so that "FF_sum_weighted" stands for the Feed Forward architecture, using the weighted sum of word vectors dataset. We applied the algorithms Adaboost, Random Forest, and Decision Tree to the mean word vectors dataset without the web category. We used OneVsRest meta-algorithm because all of these algorithms are for binary classification tasks. OneVsRest trains one classifier for every class and chooses the class label with the highest confidence.

Discussion of the results. First of all, there is a significant difference between the "web" and the "no-web" dataset. Any algorithm performs better when there is no web category. This observation leads to the conclusion that the web category is too noisy to use. The work of Fetahu, B. et al. [8] confirms our findings: users have misinterpreted the web category; some articles that clearly should be in the "news" category are in the "web" category because the resources are online.

Secondly, on the web dataset, the best performing models are the Feed Forward architectures using the averaged or summed word vectors of the dynamic Word2Vec model. To our surprise, the most straightforward architectures available outperformed the more sophisticated recurrent or convolutional approaches. The averaged or summed word vectors worked better, even though they cover less information than the raw sequence data that the other architectures use. A simple Feed Forward net might not be that sensitive to such noise since it has fewer weights to adapt.

Table 3 Corpus Statistics.

	Web dataset	Non-web dataset
Number of samples	2,470,703	1,076,097
Number of classes	4	3

Table 4. Overall results on the web dataset using dynamic Word2Vec

Model	Web	
	Accuracy	Bal. accuracy
LSTM	0.6501	0.5582
BILSTM	0.6522	0.5705
CNN	0.6239	0.4094
FF_sum	0.6657	0.5481
FF_mean	0.6687	0.5653
FF_sum_weighted	0.5768	0.3075
FF_mean_weighted	0.5816	0.3096

Table 5. Overall results without the "web" category using different word vector algorithms sorted by balanced accuracy. The dynamic Word2Vec model is abbreviated W2V, the static Google News GN and the static FastText on Wikipedia FT.

Model	Word vector model	No web	
		Accuracy	Bal. accuracy
BILSTM	W2V	0.7898	0.7664
LSTM	W2V	0.7872	0.7636
FF_MEAN	W2V	0.7806	0.7558
CNN	W2V	0.7606	0.7493
FF_SUM	W2V	0.7712	0.7450
BILSTM	FT	0.7777	0.7329
LSTM	GN	0.7697	0.7289
LSTM	FT	0.7676	0.7270
BILSTM	GN	0.7786	0.7269
FF_SUM	GN	0.7651	0.7174
CNN	GN	0.7411	0.7166
FF_SUM	FT	0.7541	0.7028
FF_MEAN	FT	0.7459	0.6955
FF_MEAN	GN	0.7634	0.6937
CNN	FT	0.7547	0.6913
AdaBoost	W2V	0.7616	0.6578
RandomForest	W2V	0.7651	0.6326
AdaBoost	GN	0.7353	0.6146
AdaBoost	FT	0.7314	0.6071
RandomForest	GN	0.7338	0.5813
RandomForest	FT	0.7300	0.5794
DecisionTree	W2V	0.7055	0.5781
DecisionTree	FT	0.6553	0.5295
DecisionTree	GN	0.6596	0.5293

The BILSTM and its unidirectional version perform best on the "no-web" dataset. They both surpass the FF models. Still, the mean word vectors models give fierce competition to the LSTM and BILSTM nets. Considering the complexities of recurrent nets

as compared to FF architectures, the difference in accuracy of 0.92% between FF_mean and BILSTM (Table 5) is remarkable. With fewer resources needed regarding disk space, GPU and GPU memory load, the Feed Forward architecture with averaged word vectors is very near to the recurrent nets. If resource consumption is not a concern, then the BILSTM would be the solution. Thirdly, the dynamic Word2Vec model outperforms the static GoogleNews and FastText models in most of the cases. The top 5 approaches all used W2V and outperformed the best static approach by more than 3%.

Lastly, the idea of using TF-IDF weighted sums or averages of word vectors failed. Two reasons might explain this phenomenon. Firstly, the semantics of word embeddings are so strong that incorporating TF-IDF adds no advantage to the classification task. Secondly, maybe because we are dealing with small documents (sentences) TF-IDF hurts the final representation of the document. However, to support our reasoning, we will need further experiments with other collections.

5 Conclusions and Future Work

In this work, we have introduced the idea of developing an intelligent service to spot citation categories automatically to enhance the quality of user-generated content at the *semantic* level. We hypothesized that the Wikipedia archive could provide a representative data source of several disciplines. Our experimental results revealed that three of Wikipedia representative citation categories do exhibit consistent patterns. We performed experiments with several algorithms to show that to a certain degree of success an intelligent service is feasible. We also confirmed that one of the current citation categories in Wikipedia requires a redefinition due to its misuse. Nevertheless, our current effort could ease the editing process in Wikipedia and any user-generated content.

References

1. Adler, B.T., de Alfaro, L.: A content-driven reputation system for the wikipedia. In: Proceedings of the 16th international conference on World Wide Web - WWW '07. pp. 261–270 (2007).
2. Bahdanau, D. et al.: Neural Machine Translation by Jointly Learning to Align and Translate. 1–15 (2014).
3. Bird, S., Loper, E.: NLTK: The Natural Language Toolkit. In: Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics. pp. 1–4 (2004).
4. Bojanowski, P. et al.: Enriching Word Vectors with Subword Information. Trans. Assoc. Comput. Linguist. 5, 135–146 (2017).
5. Brown, A.R.: Wikipedia as a Data Source for Political Scientists: Accuracy and Completeness of Coverage. PS Polit. Sci. Polit. 44, 02, 339–343 (2011).
6. Clauson, K.A. et al.: Scope, completeness, and accuracy of drug information in Wikipedia. Ann. Pharmacother. 42, 12, 1814–1821 (2008).
7. Dang, Q.V., Ignat, C.-L.: Quality Assessment of Wikipedia Articles without Feature Engineering. Proc. 16th ACM/IEEE-CS Jt. Conf. Digit. Libr. - JCDL

’16. 27–30 (2016).
8. Fetahu, B. et al.: Finding News Citations for Wikipedia. (2017).
9. Goodfellow, I. et al.: Deep Learning. MIT Press. 521, 7553, 800 (2016).
10. Gorla, N. et al.: Organizational impact of system quality, information quality, and service quality. J. Strateg. Inf. Syst. 19, 3, 207–228 (2010).
11. Graves, a et al.: Speech recognition with deep recurrent neural networks. Icassp. 3, 6645–6649 (2013).
12. Greff, K. et al.: LSTM: A Search Space Odyssey, (2016).
13. H., D.D. et al.: A general multiview framework for assessing the quality of collaboratively created content on web 2.0. J. Assoc. Inf. Sci. Technol. 68, 2, 286–308.
14. Hochreiter, S., Urgen Schmidhuber, J.: LONG SHORT-TERM MEMORY. Neural Comput. 9, 8, 1735–1780 (1997).
15. Kim, Y.: Convolutional Neural Networks for Sentence Classification. 1746–1751 (2014).
16. Kräenbring, J. et al.: Accuracy and completeness of drug information in Wikipedia: A comparison with standard textbooks of pharmacology. PLoS One. 9, 9, (2014).
17. Krizhevsky, A. et al.: ImageNet Classification with Deep Convolutional Neural Networks. Adv. Neural Inf. Process. Syst. 25. 1–9 (2012).
18. Lee, Y.W. et al.: AIMQ: A methodology for information quality assessment. Inf. Manag. 40, 2, 133–146 (2002).
19. Madnick, S.E. et al.: Overview and Framework for Data and Information Quality Research. ACM J. Data Inf. Qual. 1, 1, 1–22 (2009).
20. Manning, C.D., Raghavan, P.: An Introduction to Information Retrieval, (2009).
21. Mesgari, M. et al.: “The Sum of All Human Knowledge”: A Systematic Review of Scholarly Research on the Content of Wikipedia, (2015).
22. Mikolov, T. et al.: Efficient Estimation of Word Representations in Vector Space. Proc. Int. Conf. Learn. Represent. (ICLR 2013). 1–12 (2013).
23. Palangi, H. et al.: Deep Sentence embedding using long short-term memory networks: Analysis and application to information retrieval. IEEE/ACM Trans. Audio Speech Lang. Process. 24, 4, 694–707 (2016).
24. Royal, C., Kapila, D.: What’s on wikipedia, and what’s not... ?: Assessing completeness of information. Soc. Sci. Comput. Rev. 27, 1, 138–148 (2009).
25. Stvilia, B. et al.: Information quality work organization in Wikipedia. J. Am. Soc. Inf. Sci. Technol. 59, 6, 983–1001 (2008).
26. Sun, Y. et al.: Deep convolutional network cascade for facial point detection. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. pp. 3476–3483 (2013).
27. Zhang, Y., Wallace, B.: A Sensitivity Analysis of (and Practitioners’ Guide to) Convolutional Neural Networks for Sentence Classification. 253–263 (2015).