An end-to-end learning solution for assessing the quality of Wikipedia articles

Quang-Vinh Dang
Université de Lorraine, LORIA, F-54506
Inria, F-54600
CNRS, LORIA, F-54506
quang-vinh.dang@inria.fr

Claudia-Lavinia Ignat
Inria, F-54600
Université de Lorraine, LORIA, F-54506
CNRS, LORIA, F-54506
claudia.ignat@inria.fr

ABSTRACT

Wikipedia is considered as the largest knowledge repository in the history of humanity and plays a crucial role in modern daily life. Assigning the correct quality class to Wikipedia articles is an important task in order to provide guidance for both authors and readers of Wikipedia. The manual review cannot cope with the editing speed of Wikipedia. An automatic classification is required to classify the quality of Wikipedia articles. Most existing approaches rely on traditional machine learning with manual feature engineering, which requires a lot of expertise and effort. Furthermore, it is known that there is no general perfect feature set because information leak always occurs in feature extraction phase. Also, for each language of Wikipedia, a new feature set is required.

In this paper, we present an approach relying on deep learning for quality classification of Wikipedia articles. Our solution relies on Recurrent Neural Networks (RNN) which is an end-to-end learning technique that eliminates disadvantages of feature engineering. Our approach learns directly from raw data without human intervention and is language-neutral. Experimental results on English, French and Russian Wikipedia datasets show that our approach outperforms state-of-the-art solutions.

ACM Classification Keywords

H.5. Group and Organization Interfaces: Collaborative computing, Web-based interaction; I.7.5 Document and Text Processing: Document Analysis; I.2.6 Artificial Intelligence: Connectionism and neural nets

Author Keywords

Wikipedia, document quality, deep learning, end-to-end learning, Recurrent Neural Network (RNN), Long-Short Term Memory (LSTM)

INTRODUCTION

Wikipedia is considered as the largest knowledge repository in the history of humanity. As of this writing, Wikipedia contains about 42 million articles in all languages with 5.4 million articles belonging to English Wikipedia, as the result of the contribution from around 29 million users[1]. The size of Wikipedia increases continuously since the beginning of the website in 2001[2]. Moreover, according to Wikipedia Statistics[3], Wikipedia is being modified at an impressive speed of ten edits per second on average.

Today Wikipedia is believed as a dominant information source for an entire generation of Internet users [8]. Users tend to take for granted information on Internet such as in the domain of health-care [33]. Information presented on Wikipedia has a high impact as it is usually selected among the first results of Google search queries. Classification of Wikipedia articles into different quality classes is very important for providing guidance for readers and search engines in the selection of high quality articles and for notifying administrators and authors to improve low quality articles.

Currently, quality classes of Wikipedia articles are assigned by human reviewers[4]. These quality classes need to be reassigned after each individual modification on a given article. However, due to the very high modification speed of Wikipedia, human resource is simply not enough to review every Wikipedia revision. Naturally an automatic solution is required.

Several approaches on assessing quality of Wikipedia articles have been presented [29, 12, 14]. The ORES service[5] for quality prediction is used since 2014. Existing approaches differ in terms of classification features and learning algorithms they use such as *svm* and *random forest*. However, they share the property that the feature sets are designed manually by researchers. A feature set can be seen as a simplified model of the Wikipedia articles, a feature being defined as an individual measurable property of the process being observed [10]. Designing a good feature set is a very difficult task in machine learning [54]. In addition, the feature set is usually designed for a specific task and does not generalize well. For instance, measuring quality of Wikipedia articles in different languages requires different feature sets [26].

OpenSym '17, August 23–25, 2017, Galway, Ireland

ISBN 978-1-4503-5187-4/17/08…$15.00

DOI: https://doi.org/10.1145/3125433.3125448

[1] https://en.wikipedia.org/wiki/Wikipedia:Size_of_Wikipedia

[2] https://en.wikipedia.org/wiki/Wikipedia:Modelling_Wikipedia's_growth

[3] https://en.wikipedia.org/wiki/Wikipedia:Statistics

[4] https://en.wikipedia.org/wiki/Wikipedia:WikiProject_Wikipedia/Assessment

[5] https://mediawiki.org/wiki/ORES

In [14] we presented a preliminary work of using deep learning for automatic feature selection on assessing Wikipedia quality and achieved a promising result. However, the model needs to be retrained every-time a new article is created, which is costly in terms of time and computational resources. Therefore, it is difficult to apply this solution in practice.

There is a trade-off between accuracy, time complexity and language independence for the prediction models. In this paper, we address this trade-off and we present a deep learning approach based on Recurrent Neural Network (RNN) [49] with Long-Short Term Memory (LSTM) [28] for measuring the quality of Wikipedia articles. RNN-LSTM achieves a better accuracy compared to the state-of-the-art, is language independent and can be trained much faster than Doc2Vec. Our proposed solution is not yet applicable for real-time suggestion, but can be deployed in batch-mode and prediction can be applied at different time intervals.

Quality class assessment: problem definition

The quality class assessment problem is defined as follows. A set of quality classes is defined for a particular language of Wikipedia. A set of Wikipedia articles with quality classes manually assigned by reviewers is provided as *ground truth*. A Wikipedia article contains multiple revisions, a quality class being assigned to a particular revision. A new revision of the article is considered as a new article for which a quality class has to be assigned. The quality class assessment is a multi-class classification problem that aims to predict the quality class of a new Wikipedia article.

The quality classes defined for English, French and Russian Wikipedia datasets are provided below. The descriptions of English Wikipedia quality classes are provided in Table 1[6]. Similar definitions for French and Russian Wikipedia can be found on corresponding Wikipedia language sites.

English: FA, GA, B, C, Start, Stub.

French: ADQ, BA, A, B, BD, E.

Russian: FA, GA, SA, I, II, III, IV.

The problem we consider is measuring the quality of a given Wikipedia article, i.e. how *well* an article is written. We do not aim to measure the correctness of the information given in the article, which is a different research topic [61]. Let us consider an example. A quality classifier considers the two sentences "Bill is a man." and "Bill is a woman." similar from a quality point of view. However, a fact checking program considers these two sentences as completely different. Today, false information intentionally inserted on Wikipedia is effectively removed by vandalism detectors [24].

RELATED WORK

Several research studies on measuring the quality of Wikipedia articles have been proposed in the last decade. Except the work of [14], existing studies relied on traditional machine learning techniques with manually designed feature sets. In this section we first describe these approaches, then highlight their disadvantages and motivate the need of automatic feature engineering by means of deep learning.

Manual Feature Engineering

Approaches relying on manually defined features can roughly be divided into two groups depending on the nature of the feature set: *editor-based* and *article-based* features [53].

Editor-based features

Approaches that used editor-based information analyzed information that cannot be computed uniquely from the current content of Wikipedia pages, such as the authors of a particular article, their contributions and the duration of each contribution.

Using the hypothesis that the more reputable an author is, the higher the quality of the articles this author produces, Hu et al. [29] and Adler et al. [1] used authors *reputation* to determine the quality of Wikipedia articles. The result was confirmed in German Wikipedia [50]. The social capital of the editors could also affect the quality of the articles to which they contributed [45]. Using statistical approach, Javanmardi and Lopes [32] verified that editors reputation can be used to detect the quality of Wikipedia articles. Suzuki used *h-index* on academic ranking for assessing the quality of an article [51]. Li et al. [40] presented a modified weighted PageRank algorithm in the network of editors and articles to assess Wikipedia quality. Most recently, Betancourt et al. [6] studied the team characteristics, such as how many FA or GA articles the team members have worked on before, to predict the quality class of Wikipedia articles. These studies limit classifying FA-GA articles with other articles, but do not target classification of all quality classes as the work of [52].

Another criteria used for assessing the quality of a text is the period of time the text remains stable or is modified by other authors/reviewers. If an article has not been modified significantly for a long time, this article can be considered as mature and of high quality. For instance, Calzada and Dekhtyar [38] determined the quality of Wikipedia based on the number of *stable* articles.

Some other works presented the idea that the quality of Wikipedia articles can be determined based on the interaction between authors and reviewers [16, 58]. Wilkinson and Huberman [57] showed that a large number of authors and reviewers with an intensive cooperation should lead to high quality articles. Kittur and Kraut [37] showed that the effectiveness of adding contributors is dependent on the degree and type of coordination those contributors use. Liu and Ram [42] suggested that the behavior pattern of editors also effects articles quality.

Article-based features

The second main approach of assessing quality of Wikipedia articles is to analyze directly the content of Wikipedia articles.

One of the simplest solutions predicted the Wikipedia articles quality based on their length: the longer an article is, the better its quality [7]. This solution achieved a very high accuracy in separating between *FA* and *non-FA* articles. The approaches proposed in [41, 59] relied on editors writing styles to distinguish between *FA* and *non-FA* articles.

[6] https://en.wikipedia.org/wiki/Template:Grading_scheme

Class	Description
FA	Professional, outstanding, and thorough; a definitive source for encyclopedic information.
GA	Useful to nearly all readers, with no obvious problems; approaching (but not equalling) the quality of a professional encyclopedia.
B	Readers are not left wanting, although the content may not be complete enough to satisfy a serious student or researcher.
C	Useful to a casual reader, but would not provide a complete picture for even a moderately detailed study.
Start	Provides some meaningful content, but most readers will need more.
Stub	Provides very little meaningful content; may be little more than a dictionary definition. Readers probably see insufficiently developed features of the topic and may not see how the features of the topic are significant.

Table 1. Description of quality classes in English Wikipedia.

Dalip et al. [12] analyzed the effect of the feature set comprising text, review and network on the quality of Wikipedia articles. Authors claimed that the features that describe articles structure and style are the best to distinguish between their quality classes.

Similarly, using content, structure, network and edit history features, Anderka et al. [3] built a binary classifier to predict quality flaws in Wikipedia. The approach is based on the *cleanup tags*, which are provided by the reviewers who detected the flaws but do not have enough time or expertise to fix them.

Warncke-Wang et al. [53] presented and analyzed a feature set composed of 17 features such as article length and the number of article headings for describing the content of the Wikipedia articles. After removing some features to avoid over-fitting, the final model achieved the accuracy of 43%. The updated model using 11 features achieved the accuracy of 58% [52].

Based on the work of [52, 53], Wikimedia Foundation[7] built an online service to predict the quality class of Wikipedia articles called ORES (*Objective Revision Evaluation Service*) [25]. Currently, ORES provides predicting services for English, French and Russian Wikipedia [26]. For each language, a new feature set is required, though some features are shared[8].

Another extension of [52] was presented in [15] where we added readability scores to the feature set. We achieved the accuracy of 55%. The work is limited to English Wikipedia.

Automatic Feature Engineering

Motivation

Defining feature set is a very difficult task in machine learning [10]. A feature set can be considered as a simplified model of the given data, such as modeling a person by height, weight and date of birth. The key issue of manual feature engineering approach is information loss, i.e. some information present in the raw data but considered as irrelevant by researchers is missing in the feature set [10]. The information loss problem can be avoided if and only if the entire data is used as the feature set, as Norbert Wiener said, "the best material model of a cat is another, or preferably the same, cat" [48].

Furthermore, feature engineering mostly relies on expert knowledge which is usually expensive. Feature engineering requires effort and time, and when researchers investigate a new issue, a new feature set is needed. Andrew Ng stated that "Coming up with features is difficult, time-consuming, requires expert knowledge" [46]. Each Wikipedia language requires a new feature set which is difficult to design without a basic understanding of the language. Moreover, for a same problem and same dataset, different algorithms might require different feature sets.

Many feature selection algorithms have been proposed [10]. Their input is usually a large feature set, and their objective is to remove irrelevant features. If added to the feature set, irrelevant features do not improve or even decrease machine learning algorithm performances. No automatic method exists to define the initial feature set. In fact, the best way in practice is to add as many features as possible. A study on feature selection for text categorization [4] showed that using all features "consistently produces high and the nominally best *AUC* performance for the majority of classifiers". This study suggests using the entire document content for obtaining best performances for a text classification task.

Traditional manual feature engineering approaches rely on feature selection methods where features are gradually eliminated from an initial feature set. Unfortunately, these approaches are costly in terms of computational resources. Moreover, an infinite number of possible features can be extracted from raw data [10].

Automatic feature engineering with deep learning

Deep learning [19] can avoid manual feature engineering by learning directly from raw data. The automatic classification solution for quality of Wikipedia articles presented in [14, 13] relied on a combination of Doc2Vec [39] with deep neural networks [19]. It achieved a comparable accuracy score with state-of-the-art algorithms.

Moreover, deep learning techniques are no information loss as they take the entire Wikipedia articles as input. The study in [19] showed that as deep learning algorithms rely on the same input as human reviewers, they can achieve the same assessment performances.

The approach described in [14] can be directly applied to any language. However, this study has several shortcomings that we addressed in our proposed solution:

[7] https://wikimediafoundation.org/

[8] ORES is available at https://mediawiki.org/wiki/ORES and is under continuous development. We reported on ORES results based on [56].

- The accuracy obtained is not as good as for other state-of-the-art algorithms, such as [25].
- The model needs to be retrained each time a new article is added[9]. For avoiding retraining problem we can infer vectors of new documents by freezing the trained vectors. However, in this case, the accuracy slightly decreases.
- The solution consists of two independent phases corresponding to *Doc2Vec* and *DNN* that do not exchange any information. Therefore, there is no weight sharing [19] between the two phases which limits the performance of the deep learning model.
- The experiment has been done only on English Wikipedia.

In this study, instead of using Doc2Vec, we used Recurrent Neural Network (RNN) with Long-Short Term Memory (LSTM) to assess the quality of Wikipedia articles in end-to-end manner. Agrawal and deAlfaro [2] used LSTM to predict the quantity of user contributions by means of some manual defined features. However, they did not analyze the quality of user modifications. Graves [22] demonstrated the use of LSTM for automatically generating new Wikipedia articles based on available articles. Different from existing techniques, RNN-LSTM can be trained end-to-end without human intervention. Results of our experiments on Wikipedia datasets on different languages claim higher performance of RNN-LSTM in term of *accuracy* and *AUC* compared to state-of-the-art.

BACKGROUND KNOWLEDGE

RNN is a class of dynamic models which are used for sequence generation in different domains [22]. RNNs are defined as neural networks whose connections form cycles [21]. The model of a RNN is displayed in Figure 1.

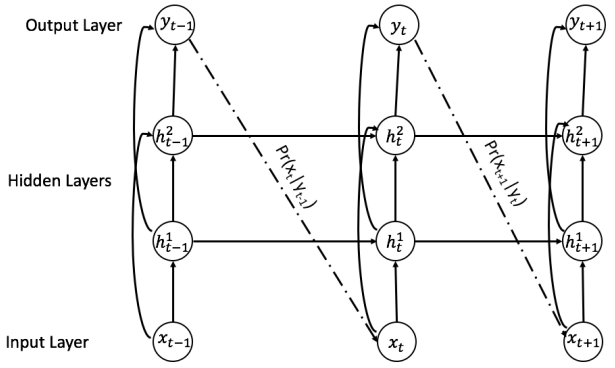

Figure 1. Deep recurrent neural network [22].

The input of an RNN consists of a sequence of tokens $X = x_1, x_2, ...x_T$ that are processed through a stack of hidden layers to compute the output y. The output is used to predict the next input token, i.e. we use the output y_t to predict the distribution of the input x_{t+1}.

The hidden layers are computed as:

$$h_t^1 = H(W_{ih^1}x_t + W_{h^1h^1}h_{t-1}^1 + b_h^1) \quad (1)$$

[9]It took several days for training Doc2Vec model on 30,000 Wikipedia articles using a cluster with 2×8-core CPU and 250GB of memory.

$$h_t^n = H(W_{ih^n}x_t + W_{h^{n-1}h^n}h_t^{n-1} + W_{h^nh^n}h_{t-1}^n + b_h^n) \quad (2)$$

wherein:

- h_t^n is the output of the n^{th} hidden layer at time t.
- W is the set of weight matrices, where W_{ih^n} is the matrix between the input and the n^{th} hidden layer, and $W_{h^jh^k}$ is the matrix that connects the j^{th} hidden layer and k^{th} hidden layer. A connection is recurrent if $j = k$.
- b is the bias vector. b_h^n is the bias vector of the n^{th} hidden layer.
- H is the activation function. Some previous works used a *sigmoid* function as activation function [11]. However, nowadays the activation function is usually defined by means of a LSTM cell.

RNN is a universal model [19] that can gradually learn any sequence at any complexity level [22]. Up to now, one of the most effective sequence models is LSTM [19]. Several studies [22, 11, 34] claimed empirically the better performance of LSTM in language modelling compared to other techniques. LSTM replaces the activation unit inside a RNN cell, which is traditionally a *tanh* activation [11]. LSTM allows the network to forget some old information while learning new knowledge. A LSTM unit is visualized in Figure 2.

The output of a LSTM cell is calculated as:

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) \quad (3)$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f) \quad (4)$$

$$c_t = f_t c_{t-1} + i_t \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \quad (5)$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_{t-1} + b_o) \quad (6)$$

$$h_t = o_t \tanh(c_t) \quad (7)$$

wherein:

- σ is the logistic sigmoid function.
- i, f, o are *input gate, forget gate* and *output gate* respectively.
- c is *memory* unit which stores the information of the LSTM cell.

After the hidden sequences h_T are computed, the output of RNN is computed as:

$$\hat{y}_t = b_y + \sum_{n=1}^{N} W_{h^ny}h_t^n \quad (8)$$

$$y_t = \gamma(\hat{y}_t) \quad (9)$$

where γ is the activation function of output layer. We used *rectifier*, which is the most popular activation function used in deep learning [9], as our activation function:

$$\gamma(x) = max(0, x) \quad (10)$$

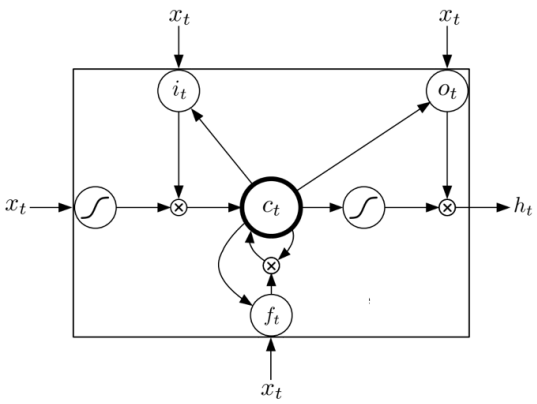

Figure 2. A Long-Short Term Memory (LSTM) cell [22].

The probability of the input sequence X is:

$$Pr(X) = \prod_{t=1}^{T} Pr(x_{t+1}|y_t) \tag{11}$$

and the loss function which is used to train the network is defined as:

$$L(X) = -\sum_{t=1}^{T} \log Pr(x_{t+1}|y_t) \tag{12}$$

In the last LSTM layer, we used a softmax cost function [9] to predict the quality class as:

$$\delta(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \text{ for } j = 1..K \tag{13}$$

We used the *cross-entropy* as the cost function to calculate the distance between the predicted value and the real value. Cross-entropy is defined as:

$$C = -\frac{1}{n}\sum_{|X|}[y * ln(a) + (1-y) * ln(1-a)] \tag{14}$$

wherein: X is the input, y is the correct output and a is the predicted value of the model. The objective of the training phase is to minimize the cost value by using back-propagation [19].

One of the major issues in classifying documents is the different length of these documents. Most machine learning algorithms are designed to work on data where all instances have the same length [54]. However, by design, RNN is perfectly fit with varied-length data because it can accumulate the learned information by its rolling back feature. In practice, among other deep architectures, RNN is often chosen for NLP tasks [19, 43], especially in language modelling [20, 18].

Originally, RNNs were designed for time-series analysis [21], so they only learn from the historical data. However, the quality of a Wikipedia article definitely depends on the entire document. Therefore, we used an extension of RNN that is bidirectional RNN [22]. The main idea of bidirectional RNN is to train two RNNs which are called *forward* and *backward* RNN. As we used RNN-LSTM, we will refer them as forward and backward LSTMs. One RNN-LSTM learns from the beginning of the document, while the other one starts from the end. Bidirectional LSTMs are more convenient because we do not have to specify the number of rollback steps [19].

While Doc2Vec learns in cross-documents manner, the advantage of RNN-LSTM is that we do not have to retrain the model when new documents are added, because it is executed on single documents. Furthermore, by using the *softmax* activation function in the last layer of the model, we can directly calculate the predicting probability of a given document to belong to a quality class [54]. On the other hand, Doc2Vec produces only vectors which represent documents, and these vectors need to be fed into a downstream machine learning algorithm [39].

EXPERIMENTS & RESULTS

Datasets

We tested our models on three Wikipedia datasets provided by Wikimedia Foundation: English, French and Russian Wikipedia. The distribution of quality classes in each dataset is provided in Table 2. The datasets are balanced, i.e. the number of articles belonging to different quality classes is similar.

	English	French	Russian
FA/ADQ/FA	4,921	1,500	1,155
GA/BA/GA	4,893	1,500	1,155
B/A/SA	4,916	1,500	1,155
C/B/I	4,908	1,500	1,155
Start/BD/I	4,913	1,500	1,155
Stub/E/III	4,917	1,500	1,155
IV			1,155

Table 2. Distribution of articles by quality class in each dataset, ranked by order of quality class from the highest to the lowest.

As described above, the input of the model is the raw text of Wikipedia articles. For instance, the raw content of Wikipedia page about Wikipedia is displayed in Figure 3.


```
'''Wikipedia''' ({{IPAc-en|audio=GT Wikipedia BE.ogg|,|w|ɪ|k|ɨ|'|p|iː|d|i|ə}} or {{IPAc-
en|audio=GT Wikipedia AE.ogg|,|w|ɪ|k|i|'|p|iː|d|i|ə}} {{respell|WIK|i|PEE|dee-ə}}) is a
free [[online encyclopedia]] that aims to allow anyone to edit articles.<ref>{{cite web
|title = Wikipedia founder defends decision to encrypt the site in China |url =
http://www.theverge.com/2015/9/4/9260981/jimmy-wales-wikipedia-china |website
= The Verge |accessdate = September 19, 2015 }}</ref> Wikipedia is the largest and
most popular general [[reference work]] on the Internet<ref name="Tancer" /><ref
name="Woodson" /><ref>{{cite
web|url=http://www.comscore.com/Insights/Press_Releases/2012/9/comScore_Medi
a_Metrix_Ranks_Top_50_US_Web_Properties_for_August_201|title=comScore MMX
Ranks Top 50 US Web Properties for August
2012|publisher=comScore|date=September 12, 2012|accessdate=February 6, 2013
}}</ref> and is ranked among the ten most popular websites.<ref name="Alexa
siteinfo" /><!-- 7th is an approximation, so sticking to vague phrasing is probably
best. --> Wikipedia is owned by the nonprofit [[Wikimedia Foundation]].<ref>{{cite
```


Figure 3. An example of raw Wikipedia text.

Implementation

We implemented our method by using tensorflow[10], an open-source deep learning library from Google. The implemented

[10] https://www.tensorflow.org/

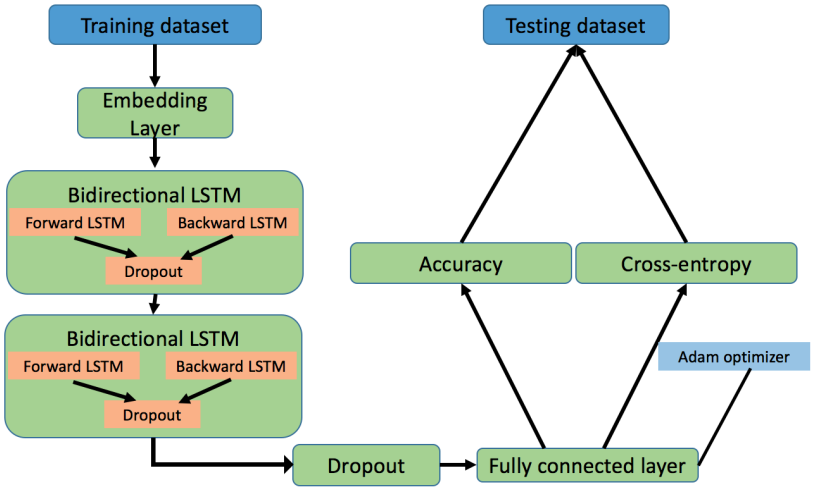

Figure 4. The bidirectional LSTM model.

program is executed on Grid5000[11]. In particular, we trained our model on several clusters, each equipped with 64GB of memory[12] and GPU Nvidia K40, Titan Black or GTX 980[13].

The hyper-parameters are optimized by using Random Discrete Search [5] using the division ratio 80/20 of the training set. The model is built with one embedding layer ($size = 20,000$ to avoid compression), two stacked LSTM layers with 512 neurons of each layer, and finally a fully connected layer ($size = 6$ as the number of quality classes). Similar with [18] we used *adaptive learning rate* with Adam optimizer [36]. The initial learning rate is set at 0.001. We used the *batch_size* of 32. The dropout ratio [60] of 0.75 is used, based on recent studies which claimed that the deep neural network models are redundant enough to apply an aggressive dropout technique [44]. In fact we tried L2 regularization with $\lambda = 1e-05$ but it does not improve the performance. We set the training epoch to 200 but in practice the model became stable after around 100 epochs. The model is visualized in Figure 4.

Validation

Similar to previous works [15], we used 5-fold cross validation for performance evaluation of the algorithms. The main idea is to divide the dataset into five equal parts, and run five times the algorithms with both training and testing phase. For each run, a single part will be used for testing while other four parts will be used for training. Using 5-fold cross validation, all the instances in the dataset will be used for testing once, so we can utilize the entire dataset for the evaluation.

Metrics

Consistently with previous research works [53, 15, 14], we used two metrics to evaluate the performance of our model: *accuracy* and *AUC* [30].

[11] https://www.grid5000.fr/

[12] By contrast, our previous model with Doc2Vec [14] cannot be executed with less than 250GB of memory.

[13] Our implementation is available at https://github.com/vinhqdang/wikipedia_analysis

Accuracy score is simply defined as:

$$accuracy = \frac{correct\_prediction\_number}{total\_prediction\_number} \quad (15)$$

Accuracy score is used because of several reasons:

- The dataset is balanced, and the *accuracy* score is the most commonly used in balanced datasets [31].
- The score is understandable even for non technical users [31]. Previously, this feature was not highly prioritized. However, recent studies in interpreting machine learning [47] emphasized the importance of letting users understand the algorithm.

However, as many studies [30] pointed out, *accuracy* does not completely characterize the performance of a classification algorithm. The score measures the final predictions of a classifier, but does not provide an understanding of algorithm behavior when the threshold changes. Therefore, we also used *ROC* (**R**eceiver **O**perating **C**haracteristic) *AUC* (***A**rea **U**nder **C**urve*) visualized in Figure 5. In this figure, TPR stands for True Positive Rate, i.e. the number of correct positive predictions divided by the total number of positive samples in testing set. FPR stands for False Positive Rate, i.e. the number of false positive predictions (the negative samples predicted as positive) divided by the total number of negative samples.

Originally, *AUC* was defined for binary classification problem. There is no standard way to extend the *AUC* calculation to multi-class classification problem. However, consistently with previous studies [15], we applied the calculation proposed by [27] which is widely used in practice. The advantage of this method is that it produces a single output value, facilitating the comparison among different algorithms.

AUC [27] is defined as follows. Given a multi-class classification problem with c classes, labelled as 0, 1, ..., $c-1$ with $c > 2$, we define $\hat{A}(i|j)$ as the probability that a randomly drawn member of class j will have a lower estimated probability of belonging to class i than a random member of class i.

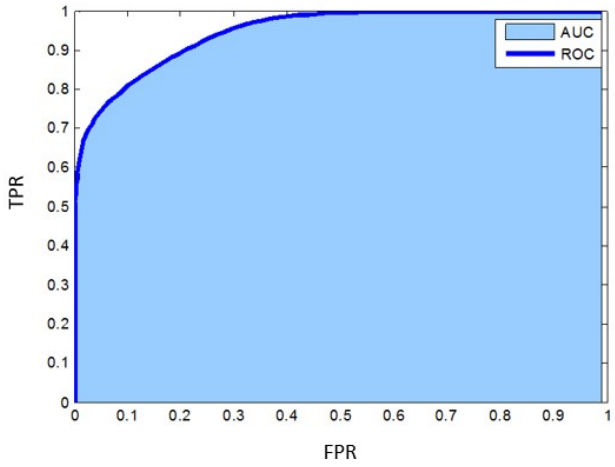

Figure 5. AUC

Then we define $\hat{A}(i,j) = [\hat{A}(i|j) + \hat{A}(j|i)]/2$. The *AUC* value is calculated as:

$$AUC = \frac{2}{c(c-1)} \sum_{i<j} \hat{A}(i,j) \tag{16}$$

Several other meta-metrics exist to measure a classifier performance, such as *F-1* or *F-2* scores. However, we did not use these scores for the following two reasons:

- The *F-scores* are originally designed for measuring performance with single-class focus [31].
- Even if we can extend *F-scores* to multi-class cases by using metrics such as *macro F-scores*, the scores only capture the performance of the classifier at a particular threshold. However, *F-scores* do not show the classifier behavior when we vary the threshold to classify the numerical outputs to corresponding classes.

For these reasons, we decided to use *accuracy* and *AUC* as reference metrics. Both *accuracy* and *AUC* values range from 0.0 to 1.0. The higher their values are, the better algorithm performance is.

Results

We compared the performance of the algorithm presented in this paper in term of *accuracy* and *AUC* with the following baseline methods:

- The method of [52], where authors used a random forest algorithm and a feature set composed of 11 features. This method is available only for English.
- The method of [15] that extends the model of [52] by adding nine features. This method is available only for English.
- The method of [14] that used Doc2Vec and DNN to classify the quality of Wikipedia articles. This method is available for all three languages.
- *ORES* method [25] that is another extension of [52] where different feature sets are defined for each Wikipedia language.

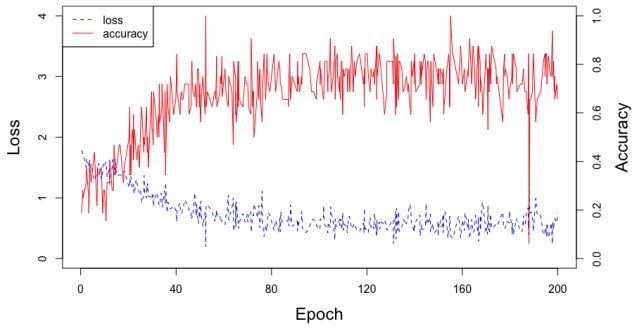

Figure 6. Loss and accuracy of training process on English Wikipedia.

The performance of the above mentioned approaches is displayed in Table 3.

We can see that the RNN-LSTM method outperforms other state-of-the-art approaches. Similar to previous works [15], the difference in performance was validated as significant with $p < 0.05$ by using McNemar test [17].

Figure 6 shows the *loss* and *accuracy* of training process for English Wikipedia of our RNN-LSTM approach. We can see that, both *loss* and *accuracy* are stable after a large number of training epochs, and *accuracy* on training data is similar with validation data. Similar graphs are obtained in French and Russian datasets. We conclude that there is no over-fitting in our RNN-LSTM approach.

DISCUSSION

The solution presented in this paper outperforms other state-of-the-art approaches and it does not require manual intervention. For instance, none of the authors understands Russian, but we can build a model to predict quality of Russian Wikipedia, which is probably not available with manual feature engineering approach. However, our proposed approach is not a "silver bullet" to solve all issues. In comparison with manual feature engineering approaches, it has a few disadvantages.

Computational time. RNN-LSTM is much slower than manual featuring approaches [15, 25]. In practice, it took around four days for training one model and six hours for testing on English Wikipedia dataset, i.e. it took around four seconds to predict the quality of one article. By contrast the methods of [15] or [25] can return the prediction for the entire testing set on the order of seconds. In contrast to Doc2Vec+DNN model [14], LSTM needs to be trained only once in RNN-LSTM approach. However, the predicting time of RNN-LSTM is too long to be implemented as a real-time service. The idea of [53] is to provide a "helper" for Wikipedia authors in writing their articles. The approaches [53, 15] can be implemented as a real-time bot, i.e. a bot can give authors a feedback as they are writing about the quality of their Wikipedia articles. We can perform the RNN-LSTM method in batch-mode, e.g. we predict the quality of a set of articles every week. Furthermore, hardware development velocity is very fast and we can expect more powerful computing devices in the future.

	English		French		Russian	
	accuracy	*AUC*	*accuracy*	*AUC*	*accuracy*	*AUC*
Warncke-Wang et al. [52]	59%	0.85	-	-	-	-
Dang & Ignat [15]	63%	0.90	-	-	-	-
Doc2Vec + DNN [14]	55%	0.79	52%	0.75	50%	0.72
ORES [25]	62%	0.86	53%	0.82	56%	0.81
RNN-LSTM	68%	0.92	65%	0.84	63%	0.83

Table 3. Performance of different algorithms

Interpretation. Interpretation is another issue in machine learning in general, referring to the difficulty of explaining the results of a machine learning model to end users [47]. While the prediction of [15, 25, 52] is somewhat understandable even for non-technical users, the results of LSTM model is a sort of cryptic. For instance, the results of [15, 25, 52] can be interpreted as suggestions such as "with two more high resolution images you can improve the quality class of your article from Start to C". On the other hand, the LSTM model provides a prediction but no explanation for this prediction. Nonetheless, studies in understanding deep learning models are receiving a lot of attention in recent years [35, 47] and we can expect a better interpretation in the future.

Noisy dataset We considered the training dataset with labelled Wikipedia articles as "ground truth". This assumption is too strong, because the quality classes are assigned by human reviewers which are subjective and noisy by nature. The noise means that, a same article might be assigned different quality classes by different reviewers. Due to this reason, we might not be able to build a perfect predictor.

CONCLUSION & FUTURE WORK

In this paper, we presented a new approach of using Recurrent Neural Networks and Long-Short Term Memory to build an end-to-end learning technique which can classify quality of Wikipedia articles without human intervention. The approach is tested against different Wikipedia languages and the performance is better than state-of-the-art.

There are several research directions which could be extended in the future. We provide here a brief guidelines of these approaches for further interests.

Adaptive RNN. Currently the hyper-parameters are tuned by random discrete search [5]. These parameters are fixed during the learning process and are independent from the input data. It should be better to build an *adaptive* RNN [23] which can automatically adjust its complexity based on the inputs.

Multi-modal modelling. In this paper, we did include the entire textual content of the documents into consideration, but we did not analyze the content of images included in Wikipedia articles. In other words, our model knows that "there is an image at this place" but it has no additional information about the image. Recently, research studies in understanding images achieved a lot of success [43] and we also plan to extend our model to capture the image content.

Transfer learning. RNN-LSTM is a supervised machine learning algorithm, which requires a lot of labelled data as training set. As labelling is done manually, the process is very costly and might contain a lot of noise. Therefore, it is difficult to apply the technique in a domain where few labelled data are available, such as Wikipedia in rare languages. The idea of transfer learning [55] is to train the model in a language where a lot of knowledge resources exist such as English Wikipedia to predict in a different language such as Vietnamese Wikipedia. The technique could save a lot of manual effort in producing labelled training dataset.

ACKNOWLEDGMENTS

The authors would like to thank Dr. Aaron Halfaker, Wikimedia Research Foundation and Morten-Warncke Wang, University of Minnesota for the datasets and the valuable discussions.

Experiments presented in this paper were carried out using the Grid'5000 testbed, supported by a scientific interest group hosted by Inria and including CNRS, RENATER and several Universities as well as other organizations (see https://www.grid5000.fr/).

REFERENCES

1. B. Thomas Adler, Krishnendu Chatterjee, Luca de Alfaro, Marco Faella, Ian Pye, and Vishwanath Raman. 2008. Assigning trust to Wikipedia content. In *WikiSym*.
2. Rakshit Agrawal and Luca de Alfaro. 2016. Predicting the quality of user contributions via LSTMs. In *OpenSym*.
3. Maik Anderka, Benno Stein, and Nedim Lipka. 2012. Predicting quality flaws in user-generated content: the case of wikipedia. In *SIGIR*. ACM, 981–990.
4. Yindalon Aphinyanaphongs, Lawrence D. Fu, Zhiguo Li, Eric R. Peskin, Efstratios Efstathiadis, Constantin F. Aliferis, and Alexander R. Statnikov. 2014. A comprehensive empirical comparison of modern supervised classification and feature selection methods for text categorization. *JASIST* 65, 10 (2014), 1964–1987.
5. James Bergstra and Yoshua Bengio. 2012. Random Search for Hyper-Parameter Optimization. *Journal of Machine Learning Research* 13 (2012), 281–305.
6. Grace Gimon Betancourt, Armando Segnine, Carlos Trabuco, Amira Rezgui, and Nicolas Jullien. 2016. Mining team characteristics to predict Wikipedia article quality. In *OpenSym*.

7. Joshua Evan Blumenstock. 2008. Size matters: word count as a measure of quality on wikipedia. In *WWW*.
8. Adam R Brown. 2011. Wikipedia as a data source for political scientists: Accuracy and completeness of coverage. *PS: Political Science & Politics* (2011).
9. Nikhil Buduma and Nicholas Locascio. 2017. *Fundamentals of Deep Learning: Designing Next-generation Machine Intelligence Algorithms*. O'Reilly Media.
10. Girish Chandrashekar and Ferat Sahin. 2014. A survey on feature selection methods. *Computers & Electrical Engineering* 40, 1 (2014), 16–28.
11. Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *CoRR* abs/1412.3555 (2014).
12. Daniel Hasan Dalip, Marcos André Gonçalves, Marco Cristo, and Pável Calado. 2009. Automatic quality assessment of content created collaboratively by web communities: a case study of Wikipedia. In *JCDL*.
13. Quang-Vinh Dang and Claudia-Lavinia Ignat. 2016a. Quality assessment of wikipedia articles: a deep learning approach. *SIGWEB Newsletter* (2016).
14. Quang-Vinh Dang and Claudia-Lavinia Ignat. 2016b. Quality Assessment of Wikipedia Articles without Feature Engineering. In *JCDL*.
15. Quang Vinh Dang and Claudia-Lavinia Ignat. 2016c. Measuring Quality of Collaboratively Edited Documents: The Case of Wikipedia. In *CIC*. IEEE.
16. Baptiste de La Robertie, Yoann Pitarch, and Olivier Teste. 2015. Measuring Article Quality in Wikipedia using the Collaboration Network. In *ASONAM*.
17. Michael Eliasziw and Allan Donner. 1991. Application of the McNemar test to non-independent matched pair data. *Statistics in medicine* 10, 12 (1991), 1981–1991.
18. Yarin Gal and Zoubin Ghahramani. 2016. A Theoretically Grounded Application of Dropout in Recurrent Neural Networks. In *NIPS*.
19. Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press.
20. Edouard Grave, Armand Joulin, Moustapha Cissé, David Grangier, and Hervé Jégou. 2016. Efficient softmax approximation for GPUs. *CoRR* abs/1609.04309 (2016).
21. Alex Graves. 2012. *Supervised Sequence Labelling with Recurrent Neural Networks*. Studies in Computational Intelligence, Vol. 385. Springer.
22. Alex Graves. 2013. Generating Sequences With Recurrent Neural Networks. *CoRR* abs/1308.0850 (2013).
23. Alex Graves. 2016. Adaptive Computation Time for Recurrent Neural Networks. *CoRR* abs/1603.08983 (2016).
24. Aaron Halfaker, Aniket Kittur, and John Riedl. 2011. Don't bite the newbies: how reverts affect the quantity and quality of Wikipedia work. In *WikiSym*.
25. Aaron Halfaker and Dario Taraborelli. 2015. Artificial intelligence service gives Wikipedians 'X-ray specs' to see through bad edits. (2015). `https://blog.wikimedia.org/2015/11/30/artificial-intelligence-x-ray-specs`
26. Aaron Halfaker and Morten Warncke-Wang. 2017. Wikipedia article quality classification (retrieved 12-Jul-2017). (2017). `https://github.com/wiki-ai/wikiclass`
27. David J. Hand and Robert J. Till. 2001. A Simple Generalisation of the Area Under the ROC Curve for Multiple Class Classification Problems. *Machine Learning* 45, 2 (2001), 171–186.
28. Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation* 9, 8 (1997), 1735–1780.
29. Meiqun Hu, Ee-Peng Lim, Aixin Sun, Hady Wirawan Lauw, and Ba-Quy Vuong. 2007. Measuring article quality in wikipedia: models and evaluation. In *CIKM*.
30. Jin Huang and Charles X. Ling. 2005. Using AUC and Accuracy in Evaluating Learning Algorithms. *IEEE Trans. Knowl. Data Eng.* 17, 3 (2005), 299–310.
31. Nathalie Japkowicz and Mohak Shah. 2011. *Evaluating learning algorithms: a classification perspective*. Cambridge University Press.
32. Sara Javanmardi and Cristina Lopes. 2010. Statistical Measure of Quality in Wikipedia. In *SOMA*.
33. S Jones. 2009. The social life of health information. *Pew research center, Washington, DC, Pew Internet & American Life Project* (2009).
34. Rafal Józefowicz, Wojciech Zaremba, and Ilya Sutskever. 2015. An Empirical Exploration of Recurrent Network Architectures. In *ICML*.
35. Andrej Karpathy, Justin Johnson, and Fei-Fei Li. 2015. Visualizing and Understanding Recurrent Networks. *CoRR* abs/1506.02078 (2015).
36. Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *ICLR*.
37. Aniket Kittur and Robert E. Kraut. 2008. Harnessing the wisdom of crowds in Wikipedia: quality through coordination. In *CSCW*. ACM, 37–46.
38. Gabriel De la Calzada and Alex Dekhtyar. 2010. On measuring the quality of Wikipedia articles. In *WICOW*.
39. Quoc V. Le and Tomas Mikolov. 2014. Distributed Representations of Sentences and Documents. In *ICML*.
40. Xinyi Li, Jintao Tang, Ting Wang, Zhunchen Luo, and Maarten de Rijke. 2015. Automatically Assessing Wikipedia Article Quality by Exploiting Article-Editor Networks. In *ECIR*.

41. Nedim Lipka and Benno Stein. 2010. Identifying featured articles in Wikipedia: writing style matters. In *WWW*.
42. Jun Liu and Sudha Ram. 2011. Who does what: Collaboration patterns in the Wikipedia and their impact on article quality. *ACM Trans. Man. Inf. Syst.* (2011).
43. Junhua Mao, Jiajing Xu, Kevin Jing, and Alan L. Yuille. 2016. Training and Evaluating Multimodal Word Embeddings with Large-scale Web Annotated Images. In *NIPS*.
44. Dmitry Molchanov, Arsenii Ashukha, and Dmitry Vetrov. 2017. Variational Dropout Sparsifies Deep Neural Networks. *CoRR* abs/1701.05369 (2017).
45. Keiichi Nemoto, Peter A. Gloor, and Rob Laubacher. 2011. Social capital increases efficiency of collaboration among Wikipedia editors. In *HT*.
46. Andrew Ng. 2013. Machine Learning and AI via Brain simulations. (2013).
47. Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *KDD*.
48. Arturo Rosenblueth and Norbert Wiener. 1945. The role of models in science. *Philosophy of science* (1945).
49. David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. 1986. Learning representations by back-propagating errors. *Nature* 323, 6088 (09 10 1986), 533–536. http://dx.doi.org/10.1038/323533a0
50. Klaus Stein and Claudia Hess. 2007. Does it matter who contributes: a study on featured articles in the German Wikipedia. In *HT*.
51. Yu Suzuki. 2015. Quality Assessment of Wikipedia Articles Using *h*-index. *JIP* 23, 1 (2015), 22–30.
52. Morten Warncke-Wang, Vladislav R. Ayukaev, Brent Hecht, and Loren G. Terveen. 2015. The Success and Failure of Quality Improvement Projects in Peer Production Communities. In *CSCW*.
53. Morten Warncke-Wang, Dan Cosley, and John Riedl. 2013. Tell me more: an actionable quality model for Wikipedia. In *OpenSym*.
54. Jeremy Watt, Reza Borhani, and Aggelos Katsaggelos. 2016. *Machine Learning Refined: Foundations, Algorithms, and Applications*. Cambridge Univ. Press.
55. Karl Weiss, Taghi M Khoshgoftaar, and DingDing Wang. 2016. A survey of transfer learning. *Journal of Big Data* 3, 1 (2016), 1–40.
56. Wikimedia. 2017. Objective Revision Evaluation Service (retrieved 24-Apr-2017). (2017). https://meta.wikimedia.org/wiki/Objective_Revision_Evaluation_Service/wp10
57. Dennis M. Wilkinson and Bernardo A. Huberman. 2007. Cooperation and quality in Wikipedia. In *WikiSym*.
58. Guangyu Wu, Martin Harrigan, and Pádraig Cunningham. 2012. Classifying Wikipedia articles using network motif counts and ratios. In *WikiSym*.
59. Yanxiang Xu and Tiejian Luo. 2011. Measuring article quality in Wikipedia: Lexical clue model. In *SWS*.
60. Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. 2015. Recurrent Neural Network Regularization. *CoRR* abs/1409.2329 (2015).
61. Yudian Zheng, Guoliang Li, Yuanbing Li, Caihua Shan, and Reynold Cheng. 2017. Truth Inference in Crowdsourcing: Is the Problem Solved? *PVLDB* 10, 5 (2017), 541–552.