WikiWrite: Generating Wikipedia Articles Automatically

Siddhartha Banerjee[†]
[†]The Pennsylvania State University
University Park, PA, USA
sbanerjee@ist.psu.edu

Prasenjit Mitra[†*]
[*]Qatar Computing Research Institute
HBKU, Doha, Qatar
pmitra@ist.psu.edu

Abstract

The growth of Wikipedia, limited by the availability of knowledgeable authors, cannot keep pace with the ever increasing requirements and demands of the readers. In this work, we propose WikiWrite, a system capable of generating content for new Wikipedia articles automatically. First, our technique obtains feature representations of entities on Wikipedia. We adapt an existing work on document embeddings to obtain vector representations of words and paragraphs. Using the representations, we identify articles that are very similar to the new entity on Wikipedia. We train machine learning classifiers using content from the similar articles to assign web retrieved content on the new entity into relevant sections in the Wikipedia article. Second, we propose a novel abstractive summarization technique that uses a two-step integer-linear programming (ILP) model to synthesize the assigned content in each section and rewrite the content to produce a well-formed informative summary. Our experiments show that our technique is able to reconstruct existing articles in Wikipedia with high accuracies. We also create several articles using our approach in the English Wikipedia, most of which have been retained in the online encyclopedia.

1 Introduction

Wikipedia extensively covers articles on many topics; however, not all categories on Wikipedia have enough articles. The number of red-linked articles[1], i.e., entities that are referred to from other articles but have not actually been authored, keeps increasing. A list of articles that are red-linked from over 20 or more other Wikipedia articles exists[2].

Existing methods [Sauper and Barzilay, 2009; Banerjee *et al.*, 2014] generated Wikipedia articles that needed substantial edits from human editors. In this work, we attempt to address three drawbacks of these works. First, all existing methods for Wikipedia article generation assume that the Wikipedia categories[3] are known. Furthermore, even if the category information is known, articles often belong to multiple categories that are often not equally important. Learning from several categories may result in a mish-mash of sections from different categories being chosen resulting in a less polished article. In some cases, categories do not provide much information, for example, *1959 births* or *Living people* in *Octavio Solis*[4] article. The second issue, copyright violations, imply that content on the entity retrieved from the web cannot be directly copied into Wikipedia [Banerjee *et al.*, 2014]. To tackle the issue of copyright violation, we proposed an abstractive summarization system [Banerjee and Mitra, 2015c] using sentence fusion to generate novel sentences before augmenting content into Wikipedia stubs. Despite our efforts, several copyright violations were reported because the system simply used the original words from the source documents without any *paraphrasing*. The third issue arises because we choose information from multiple documents. Sentences selected (and paraphrased) from multiple documents must be ordered such that the resulting article is coherent. Existing summarization systems for Wikipedia did not tackle *coherence*. Balancing all the above mentioned factors simultaneously in a single end-to-end system is hard. To the best of our knowledge, these three issues have not been addressed by any existing Wikipedia article generation system.

In this work, we propose **WikiWrite**, a system to author new articles on Wikipedia automatically. We obtain the vector representations of the red-linked entities using a paragraph vector model [Le and Mikolov, 2014] that computes continuous distributed vector representations of varying-length texts. Subsequently, we identify existing Wikipedia articles that are semantically close (or similar) to the red-linked entity in this vector space using cosine similarity. We emulate the structure of similar articles to construct the article for a red-linked entity. We identify the important sections and assign relevant web-content (assuming information is available) to the sections. To the best of our knowledge, our work on automatic Wikipedia article construction is the first to learn content templates from similar articles *without requiring any information on Wikipedia categories* and thus, is fundamentally different from all the other work done in the area of

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

[2] https://en.wikipedia.org/wiki/Wikipedia:Most-wanted_articles

[3] https://en.wikipedia.org/wiki/Help:Category

[4] https://en.wikipedia.org/wiki/Octavio_Solis – page as of 27th January' 2016

automatic Wikipedia article construction. Furthermore, our model jointly optimizes the ordering of sentences (coherence) along with the informativeness and linguistic quality of the summary (of the content assigned to the sections in the article). We compute the coherence score between any two sentences using transition probabilities of word-pairs (nouns and verbs) between the sentences. The transition probabilities are learned from pairs of adjacent sentences that exist in the similar articles. We also propose an optimization model to find a suitable set of lexical and phrasal transformations for paraphrasing the generated summaries.

We conduct multiple experiments to evaluate the efficiency of our proposed technique. First, we compare the accuracies of our section assignment task with other comparable systems. Second, we reconstruct existing articles on Wikipedia by retrieving content from the web and compare the content in the system generated articles and the actual articles on Wikipedia. Third, we create 50 new articles in the English Wikipedia. The results of our experiments suggest that **WikiWrite** can generate basic versions of new articles reasonably well. More than 90% of the automatically generated articles on Wikipedia have been retained, often with minor changes made by the reviewers. Our classifier works significantly better than other comparable classifiers (~29% improvement) in assigning content to relevant sections.

2 Related Work

Previous work on Wikipedia content generation relied heavily on manually annotated information of Wikipedia categories [Sauper and Barzilay, 2009; Banerjee and Mitra, 2015c]. Articles in Wikipedia consist of sections. Sauper and Barzilay [2009] retrieved content from the web on articles belonging to the category of diseases by using the most frequent section titles as keywords to retrieve relevant web-snippets (excerpts). The most informative excerpts were selected using a perceptron-based framework and populated into the Wikipedia article. In a recent work, Banerjee and Mitra [Banerjee and Mitra, 2015c; 2015b; 2015a] proposed WikiKreator where contents in the Wikipedia sections were represented by topic-distribution features using Latent Dirichlet Allocation (LDA) [Blei *et al.*, 2003]. The features were used to train classifiers and predict sections for new content retrieved from the web. However, the number of classes were restricted to only the 10 most frequent sections. Limiting to only a few important sections in a category resulted in missing out on appending relevant sections[5] to the stubs. In contrast to previous work, our approach does not require information on Wikipedia categories.

In our prior work, we proposed an abstractive summarization framework [Banerjee *et al.*, 2015] to summarize news articles and extended that system to populate new information in stubs on Wikipedia. The summarization framework used an integer-linear programming (ILP) formulation that optimizes informativeness and linguistic quality jointly. However, editors removed content appended by WikiKreator on grounds of incoherence. Copyright violation issues were also reported where long sequences of text were retained from the original sources. Recent work on summarization [Parveen and Strube, 2015] has focused on integrating various components (importance, non-redundancy and coherence) for single document *extractive* summarization. In contrast, we propose an integrated framework for multi-document *abstractive* summarization.

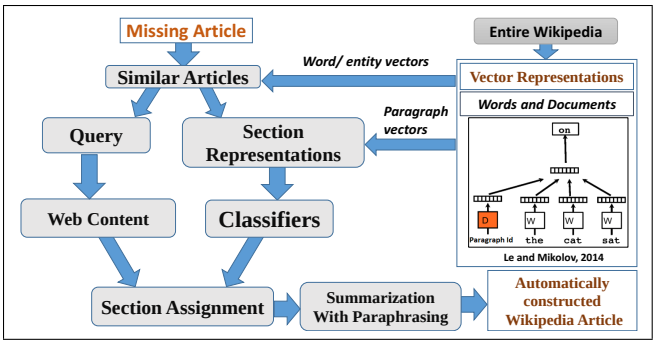

Figure 1: WikiWrite: Our Proposed Framework

3 Proposed Approach

Our goal is to construct articles for notable[6] Wikipedia entities that do not have corresponding articles. Figure 1 shows our proposed framework of **WikiWrite**. As can be seen from the figure, we use the entire Wikipedia to obtain D dimensional representations of words/entities[7] as well as documents using the paragraph vector distributed memory (PV-DM) model [Le and Mikolov, 2014]. Similar articles are identified using cosine similarity between the vector representations of the missing entity and representations of the *existing entities* (entities that have corresponding articles). Content from the similar articles are used to train multi-class classifiers that can assign web-retrieved content on the red-linked entity to relevant sections of the article. Furthermore, we propose a novel two-step ILP-based paraphrastic summarization technique to generate short, concise and paraphrased summaries for each section in the Wikipedia article.

3.1 Entity Representations

Entities that are similar are mentioned in similar contexts in other articles on Wikipedia. For example, the entity *Sonia Bianchetti* did not have a corresponding article in the English Wikipedia (as of November, 2015). However, it has been mentioned (red-linked) in 28 other articles in the context of words such as *referee, International Skating Union (ISU), Judge*, etc. Articles similar to *Sonia Bianchetti* predicted by the **PV-DM** model are *Elemér Terták, Lawrence Demmy, Josef Dědič, etc.*, all of whom happen to be referees/ skaters under ISU. Therefore, we can create an article on "Sonia Bianchetti" by emulating the structure from the above mentioned similar articles.

[5] For example, *Society and culture* is an important section in the article on *Parkinson's disease*; however, it is not one of the most frequent sections in the articles within the diseases category on Wikipedia.

[6] https://en.wikipedia.org/wiki/Wikipedia:Notability

[7] In our approach, we consider each multi-word entity such as *New York City* as a single word to effectively compute their vector representations.

PV-DM: The PV-DM model is based on the principle that several contexts sampled from the paragraph can be used to predict the next word. Given a sequence of T words $(w_1, w_2, w_3, ..., w_T)$, the task is to maximize the average log probability. In the equation 1, c is the size of the context (number of words before and after the current word to be used for training). The conditional probability of w_{t+j} given w_t is given by the softmax function [Bridle, 1990] in equation 2, where $v_{w_{t+j}}$ and v_w refers to the output and the input vector representations of the word w, respectively. W refers to the total number of words in the vocabulary.

$$F = \frac{1}{T}\sum_{t=1}^{t=T}\sum_{-c\leq j\leq c, j\neq 0} \log p(w_{t+j}|w_t) \tag{1}$$

$$p(w_{t+j}|w_t) = \frac{\exp({v'_{w_{t+j}}}^T v_{w_t})}{\sum_{w=1}^{W} \exp({v'_w}^T v_{w_t})} \tag{2}$$

We used the implementation of PV-DM from the gensim *doc2vec* package [Řehůřek *et al.*, 2011] to generate D dimensional vector representations for both the words and the paragraphs of text simultaneously. The PV-DM model serves two main purposes: (i) Identification of similar articles on Wikipedia and (ii) Inference of vector representations of new paragraphs retrieved from the web.

3.2 Section Content Generation

The content generation stage consists of two steps. First, we need to retrieve informative pieces of text from the web that are relevant to the entity and assign them into the appropriate sections in the article. Second, we need to effectively summarize and rewrite the assigned content.

Information retrieval

We employ **query reformulation** [Aula, 2003] to augment keywords to the entity and create a modified query to retrieve relevant documents from the web. The introductory sentence of similar Wikipedia articles should contain similar keywords that characterize the entities. Nouns have been found to characterize domain specific texts surprisingly well [Riloff and Lorenzen, 1999]. Therefore, we extract the two most frequent nouns from the introductory sentences of the 20 similar articles as relevant keywords. To create an article on "Machine learning", the reformulated query – *"Machine Learning" algorithm intelligence* (with quotes for exact matching of article entity), is obtained by appending the two most frequent nouns in the similar articles (*algorithm* and *intelligence*) along with the entity name. Finally, we retain only the informative content (excerpts) using boilerplate detection [Kohlschütter *et al.*, 2010] from the top 20 search results from Google using the reformulated query. We assign each excerpt into the relevant sections using a text classifier trained using content from the articles similar to the red-linked entity. The section titles from the articles represent the classes and the corresponding textual content in the sections (their PV-DM representations as features) as the instances. However, semantically similar sections might have different titles. For example, articles in the *Diseases* category on Wikipedia contain sections on *Symptoms* as well as *Signs and Symptoms*. Both these sections should be assigned to the same class by a text classifier. To tackle such cases, we apply repeated bisection clustering (RBR) [Zhao and Karypis, 2002] using PV-DM features with the section titles as the labels. Finally, we retain only clusters that have an intra-cluster similarity of at least 0.5. We set the most common section title in each cluster as the class label of that cluster.

Paraphrastic Summarization

We refer to our content summarization approach as *paraphrastic* because we include an additional step on rewriting (paraphrasing). First, an ILP model is proposed that selects and constructs a summary by maximizing coherence, informativeness and linguistic quality simultaneously. Thereafter, we propose a paraphrasing step to determine the most optimal set of lexical and phrasal transformations to rewrite sentences in the summary.

Novel sentence generation: We follow a word-graph construction approach [Filippova, 2010] to generate new sentences. Our prior work [Banerjee and Mitra, 2015c] also displayed the effectiveness of generating such sentences using shallow syntactic features based on words and their parts-of-speech (POS) tags. However, to improve grammaticality, we use bigrams instead of unigrams. The bigrams in the sentences represent the nodes in the word-graph. Edges are constructed between the nodes if the first word in the bigrams are adjacent in any of the sentences. Traversing paths along the graph generates new sentences. To create true abstractive summaries, generated sentences that are very similar to the original sentences (cosine similarity $\geq$ 0.80) are discarded. Furthermore, not all generated sentences are grammatical. Therefore, the next step is to select only a few of the generated sentences in the summary using our ILP formulation.

ILP Formulation: We formulate the following ILP objective function as shown in Equation 3.

$$F = \sum_{i=1}^{K} w^{p_i} \cdot p_i + \lambda \sum_{a_{i,j}\in A} coh_{i,j} \cdot arc_{i,j} \tag{3}$$

$$w^{p_i} = I^{p_i} \cdot Sim^{intra}(p_i) \cdot LQ(p_i) \tag{4}$$

Let us assume that our bigram graph-based technique generates K new sentences. The binary variables p_i ($\forall i \in [1 \dots K]$) represent the generated sentences while $arc_{i,j}$ ($\forall i \in [1 \dots K], j \in [1 \dots K], i \neq j$) represents the transitions between the sentences. w^{p_i}, as shown in equation 4, is the total contribution of the generated sentence p_i computed using the importance of the sentence (I^{p_i}), average intra-sentence similarity ($Sim^{intra}(p_i)$) and the linguistic quality ($LQ(p_i)$). We use the cosine similarity between the sentence and the reformulated query to compute the importance of a sentence. We compute linguistic quality using a trigram language model score [Banerjee *et al.*, 2015]. Therefore, Sentences that are generated using similar initial sentences are assigned higher weights using ($Sim^{intra}(p_i)$), which computes the average of the pairwise cosine similarities of the sentences used to generate p_i. $coh_{i,j}$ denotes the coherence between two sentences p_i and p_j. The variable $arc_{i,j}$ should be 1 if sentence p_i should precede p_j in the final summary,

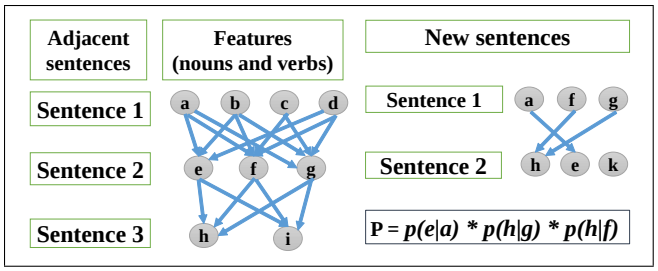

Figure 2: Local coherence estimation between sentences

otherwise it is 0. λ is a scaling parameter that is set to 10 empirically.

Coherence ($coh_{i,j}$): Text coherence has been studied vastly in the NLP literature [Lapata and Barzilay, 2005]. We assume that the global coherence of a paragraph is a combined effect of the individual local coherences (coherence between adjacent sentences). We compute local coherence by estimating transition probabilities of the features from one sentence to another following Lapata's probabilistic structuring approach [Lapata, 2003]. Local coherence can be determined by multiplying the individual transition probabilities of the features in the adjacent sentences. In this work, we restrict the features to the set of nouns and verbs in the sentences. For example, given three adjacent sentences as shown in figure 2, we consider all the possible combinations of the features (alphabets *a* to *i*). For example, when there is a transition from sentence 1 to 2, we consider combinations *a* $\rightarrow$ *e,f,g; b* $\rightarrow$ *e,f,g*, etc. In the entire set of the similar articles, we compute the total frequency of transitions from *a* $\rightarrow$ *e* as well as *a* $\rightarrow$ *all other features*. The transition probability of *a to e* is simply computed as the ratio of the frequency of the transition from *a* to *e* to the frequency of *a* to all the features. The final coherence score between two sentences is equal to the product of the individual transition probabilities of the feature combinations.

We impose a number of constraints to solve the optimization problem. First, we avoid redundancies between the sentences in the summary and thereby introduce constraint 5. If two sentences p_i and p_j have high cosine similarity ($\geq$0.5), only one out of the two sentences can be included in the summary. Equations 6 constrain the arcs (variables that denote order between the sentences) to generate a coherent summary. The formulations to constrain the arcs have been adapted from Nishikawa et al.'s [Nishikawa *et al.*, 2010] prior work on opinion summarization. An arc between two sentences can only exist in the final summary if both the sentences connected by the arc are also included in the summary. We introduce two dummy sentences, p_s and p_e (they are always 1 as first and last sentences should always exist) that denote the starting and the ending points of the summary. Consequently, dummy arcs are also created that connect the starting and ending sentences. Therefore, if $arc_{s,i}$ is 1, the final summary should have sentence p_i as the first sentence in the generated summary. There should be only one starting and one ending arc to account for the first and the last sentence in the summary. Furthermore, there should be just one incoming arc and one outgoing arc from each sentence if the sentence is selected in the summary.

$$\forall i, j \in [1 \dots K]\ i \neq j, p_i + p_j \text{ if } sim(p_i, p_j) \geq 0.5. \quad (5)$$

$$\sum_i arc_{s,i} = 1 \quad \sum_i arc_{i,e} = 1 \quad \sum_i arc_{i,j} = \sum_i arc_{j,i} \forall j$$
$$\sum_i arc_{i,j} + \sum_i arc_{j,i} = 2p_j \forall j \quad (6)$$

$$\sum_{i=1}^{K} l_i p_i \leq L_{max} \quad (7)$$

Equation 7 is used to limit the summary in each section to a maximum of L_{max} words. To avoid cycles in the final arc selection, we use the constraints as used for headline generation in earlier work [Deshpande *et al.*, 2007]. Th final sentences represented by the solution of the ILP are ordered based on the final state of the $arc_{i,j}$ variables. The sentence that has the highest similarity (I^{p_i}) with respect to the reformulated query is used as the introductory sentence in the article.

Sentence Rewriting: We rewrite each sentence by applying modifications to words and phrases in the sentences. First, we identify the candidate set of possible modifications that can be used to rewrite a sentence from the Paraphrase database (PPDB) [Ganitkevitch *et al.*, 2013]. The set of possible modifications T for a sentence can be denoted as $\{t_1, t_2.t_3, \dots, t_{|T|}\}$. For example, two possible modifications are possible for the sentence: *The NSSP initiative will lead to significant economic benefits for both countries.*

- *significant economic => considerable economic*
- *economic benefits => financial advantages*

We assign a readability score to each transformation using a trigram language model score of the textual content within a window size of 2 words in either direction of the modification. For example, the readability score (LQ), for the first modification can be computed using the sequence – "lead to ***considerable economic*** benefits for". To determine the best possible transformations, we maximize the following objective function:

$$R(t_1, \dots, t_{|T|}) = \sum_{i=1}^{|T|} Sim(seq_o, seq_i) \cdot LQ(seq_i) \cdot t_i \quad (8)$$

where seq_i and seq_o refer to the sequence of words after and before applying the i^{th} transformation, respectively. $Sim(seq_o, seq_i)$ computes the cosine similarity between the semantic representation (using PV-DM) of the old and new sequences. The parameter $Sim(seq_o, seq_i)$ prevents excessive deviation in meaning from the original sentence during rewriting. We also constrain overlapping modifications using equation 9. For example, the word *economic* in the example can be transformed using only one of the competing modifications.

$$\forall j, j' \in [1, |T|], j \neq j', t_j + t_{j'} \leq 1 \text{ if } intersect(t_j, t_{j'}) > 0 \quad (9)$$

The $intersect$ function checks if the two modifications t_j and $t_{j'}$ affect any common words or not. Only one of the overlapping modifications can be applied. The above sentence, is therefore paraphrased as:

The NSSP initiative will result in major financial advantages for the two countries.

4 Experimental Results

We conducted several experiments to evaluate the efficiency of **WikiWrite**. First, we evaluate **WikiWrite**'s accuracy in assigning content to relevant sections. Second, we evaluate the quality of the articles generated by **WikiWrite**. Third, we generate new articles (for the red-linked entities) on Wikipedia using our approach and measure the retention rate[8] of the content.

Data Characteristics: Wikipedia regularly dumps all the data in regular intervals. We used the dump dated 2nd June, 2015[9] in our experiments. The total size of the corpus (only article contents) was close to 50 GB (~12 GB compressed). We limit to articles that contained at least 20 words. Our corpus finally contained 4.8 million articles. Furthermore, the corpus contained 15,500 red-linked entities that were referenced at least 20 times from other articles.

Baselines: We compare the performance of **WikiWrite** with two existing systems for Wikipedia article generation. The first baseline, **Perceptron-ILP**, proposed by Sauper and Barzilay [2009], uses a perceptron-based ranking algorithm to select informative excerpts in the article. The second baseline, **WikiKreator** [Banerjee and Mitra, 2015c], was primarily designed to improve stubs[10]. The query was reformulated by extracting keyphrases from the introductory sections of the articles. However, the introductory content is not available when generating completely new articles. Therefore, to have a fair comparison, we provide the same excerpts (from web sources) to both **WikiKreator** and **WikiWrite**.

Both the baselines require information on the Wikipedia categories. Specifically, content from comprehensive articles[11] within the entire category are used to train the systems. Selection of a single category among a list of categories (that each article is assigned to) is particularly challenging. Therefore, we use the comprehensive articles from all the categories to train the baselines and select only the most frequent sections (top 10) from the entire set of articles.

Parameters: We use 100 dimensional vector representations (parameter D) for the entities and paragraphs of text[12]. For the classification task (assigning content into relevant sections in the article), we experimented with several machine learning classifiers (Random Forest, Naive-Bayes and Support Vector Machines). Random Forest (RF) [Breiman, 2001] performed the best in our classification task on existing Wikipedia articles and hence we report only the results using RF. L_{max}, the maximum number of words in each section is dynamically set to the average number of words in the sections that were clustered together using RBR(see 3.2).

4.1 Reconstructing existing articles

Reconstructing articles using the baseline systems consume significant amount of time because of the requirement of learning from all articles within a category. Tasks such as preprocessing of the articles, topic models (in **WikiKreator**), etc. are resource intensive. Therefore, to ensure the repeatability of such experiments, we restrict to 1000 randomly selected fairly popular articles (that are mentioned at least 20 times in other articles) for the reconstruction experiment. The content in these 1000 articles were not used in training in any of the systems. We evaluate two different aspects – (i) **Classification:** Assigning content into appropriate sections and (ii) **Content Selection:** Retaining important content from the web sources in the final article.

Table 1: Section Classification Results

Technique	F1-score	Average Time
WikiWrite	0.622	~2 mins
WikiKreator	0.481	~10 mins

Classification: The classification task implies prediction of the section title given the content in the section. The contents in the sections of the randomly selected articles are considered to be the instances. We consider the section titles (labels) on Wikipedia to be the ground truth. In **WikiKreator**, training is performed using articles within all the categories as mentioned in the actual article on Wikipedia. On the contrary, no information on categories is required for **WikiWrite**. The classifier (RF) learns from 500 most similar articles computed using cosine similarities of the vector representations (from PV-DM). We do not compare with **Perceptron-ILP** for the classification task as the system does not involve any classification.

As can be seen from the table 1, we compute the average F1-scores and also show the average time required to classify the sections in each article. **WikiWrite** outperforms **WikiKreator** on this classification task as can be seen from the F1-scores. **WikiWrite** is able to assign content to sections more effectively by learning from similar articles rather than restricting to only the most frequent sections in the categories. Furthermore, **WikiWrite** is able to assign the sections significantly faster (~5X times) than **WikiKreator** (LDA [Blei *et al.*, 2003] requires significant time to run).

Content Selection: We also evaluate the proportion of the Wikipedia articles that can be constructed using knowledge from the web. The content in the existing articles on Wikipedia is considered as the ground truth. We construct articles for the same 1000 randomly selected Wikipedia entities as described earlier. To evaluate the effectiveness of our query reformulation technique, we also create a modified system **(WikiWrite (Ref))** that uses only the references listed in the Wikipedia article to reconstruct the article. We evaluate the article content using ROUGE [Lin, 2004]. ROUGE scores have been found to be useful in measuring summarization quality and also has been used in earlier Wikipedia article generation evaluation. We report ROUGE-1 (unigram matches) and ROUGE-2 (bigram matches) recall scores in our results. To account differences in article length, we restrict ROUGE comparison to the first 200 words in each section. The length constraints for summary generation in both the systems is set to 200 words, accordingly.

As can be seen from the table 2, **WikiWrite** outperforms both **WikiKreator** and **Perceptron-ILP** according to both the ROUGE scores. **WikiWrite (Ref)** performs better than

[8] Percentage of generated articles retained on Wikipedia

[9] https://archive.org/details/enwiki-20150602

[10] https://en.wikipedia.org/wiki/Wikipedia:Stub

[11] Articles that are not tagged as stubs on Wikipedia.

[12] We run all experiments on a computer with i7 processor and 16 GB RAM. Setting $D = 100$ requires roughly 10 GB of memory.

Table 2: Content Selection Results

Technique	ROUGE-1	ROUGE-2
WikiWrite	0.441	0.223
WikiWrite (Ref)	0.520	0.257
WikiKreator	0.371	0.183
Perceptron-ILP	0.342	0.169

Table 3: Statistics of Wikipedia stub content addition

Statistics	WikiKreator	WikiWrite
Number of stubs appended	40	40
Entire edit retained	15	32
Modification of content	5	5
Content Removed	20	3
Avg. change in size	287 bytes	424 bytes
Avg. no of edits	3.82	1.39

WikiWrite because we use more reliable and verified references from the Wikipedia articles. All the systems except **Perceptron-ILP** consist of a summarization component. Therefore, even with the same or similar web sources, the systems equipped with summarizers retain more informative (higher ROUGE scores) content. **Perceptron-ILP**, on the contrary, generates articles using entire excerpts obtained from the web in each section. As a result, ROUGE scores obtained using **Perceptron-ILP** are lower than the other systems. Furthermore, using entire excerpts currently makes it ineligible for Wikipedia article generation because of copyright violation regulations.

4.2 Generating Non-existing content in Wikipedia

We can compare different systems by generating articles using all the techniques and comparing the qualities of the articles on Wikipedia. However, adding articles with subpar content is not justifiable for the readers on Wikipedia. Therefore, we do not generate articles on Wikipedia using all the techniques. However, evaluating content addition to stubs is often easier than article generation. The stubs are already in the Wikipedia mainspace and are constantly monitored. In contrast to stubs, drafts of new articles on Wikipedia require adequate time (often weeks)[13] before they are moved into (or deleted from) the Wikipedia mainspace. We compare the summarization algorithms of **WikiWrite** and **WikiKreator** by adding content generated by both systems to randomly selected stubs.

Table 3 shows the results of our experiments of adding content to stubs using both the techniques. As can be seen from the table, content appended using our paraphrastic summarization technique has been retained in more cases (80%) than **WikiKreator** (∼38%) that uses source words from the documents. The results clearly show that paraphrasing plays a very important role in avoiding issues of plagiarism and grammaticality.

We also constructed 50 randomly selected articles using WikiWrite that did not exist in Wikipedia. Table 4 shows the current statistics[14] of recently generated articles that have been automatically created using **WikiWrite**. As can be seen, 47 articles out of the 50 have been moved into the Wikipedia mainspace. The high retention rate of the articles shows that **WikiWrite** can generate Wikipedia articles of reasonable quality automatically. The entire content added by our approach was retained only in 12 articles, all other articles underwent at least a few edits. In some cases, the edits were mostly used to resolve minor references. For example, the article on *Dick Barbour*[15] that we created using automatically generated content has been retained with a minor change. However, in certain cases, multiple edits (by other authors) were made to improve overall quality of the article. For example, *Atripliceae*[16] has undergone extensive modification by authors from its initial state. In certain cases, reviewers reported issues related to the language and the tone of the articles, for example, *Talonid*[17]. As we are continuously improving our bot, our initial entries in a few cases contained syntactical Wiki mark-up errors, which we resolved manually. The edit history in such articles will show up as multiple edits made by us. Three generated articles were removed on grounds of using promotional (self-advertising) content. In summary, **WikiWrite** generated content has been mostly retained on Wikipedia and is undergoing modifications everyday to continuously improve the quality of the articles.

Table 4: Statistics of Wikipedia article generation

Statistics	
Number of articles in mainspace	47
Entire edit retained	12
Modification of content	35
Average number of edits	11
Percentage of references retained	72%

5 Conclusions and Future Work

In this work, we proposed, WikiWrite, a system that can automatically create new Wikipedia articles. We used an existing framework to represent varying length texts on Wikipedia as continuous vector distributions. The representations are used to identify similar articles that currently exist on Wikipedia. Classifiers are trained using content from the similar articles to assign web content into the relevant sections in the new article. We also propose a novel summarization technique that handles coherence of sentences and also applies paraphrasing. Our proposed summarization technique outperforms other comparable methods for Wikipedia content generation because of its ability to coherently summarize and rewrite content retrieved from the web. Articles generated using our system can be a good starting point for authors, who can modify and polish the content to make it fit for the online encyclopedia. We generated multiple articles using WikiWrite and most of the articles have been successfully retained. In the future, we will develop techniques to rewrite sentences to meet the requirements of encyclopedic tone of articles on Wikipedia.

[13] https://en.wikipedia.org/wiki/Wikipedia:Articles_for_creation

[14] Statistics as of January 25th, 2016

[15] https://en.wikipedia.org/wiki/Dick_Barbour

[16] https://en.wikipedia.org/wiki/Atripliceae

[17] https://en.wikipedia.org/wiki/Talonid

References

[Aula, 2003] Anne Aula. Query formulation in web information search. In *ICWI*, pages 403–410, 2003.

[Banerjee and Mitra, 2015a] Siddhartha Banerjee and Prasenjit Mitra. Filling the gaps: Improving wikipedia stubs. In *Proceedings of the 2015 ACM Symposium on Document Engineering*, pages 117–120. ACM, 2015.

[Banerjee and Mitra, 2015b] Siddhartha Banerjee and Prasenjit Mitra. Wikikreator: automatic authoring of wikipedia content. *AI Matters*, 2(1):4–6, 2015.

[Banerjee and Mitra, 2015c] Siddhartha Banerjee and Prasenjit Mitra. Wikikreator: Improving wikipedia stubs automatically. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pages 867–877, 2015.

[Banerjee *et al.*, 2014] Siddhartha Banerjee, Cornelia Caragea, and Prasenjit Mitra. Playscript classification and automatic wikipedia play articles generation. In *Proceedings of the 22nd International Conference on Pattern Recognition (ICPR)*, pages 3630–3635. IEEE, 2014.

[Banerjee *et al.*, 2015] Siddhartha Banerjee, Prasenjit Mitra, and Kazunari Sugiyama. Multi-document abstractive summarization using ilp based multi-sentence compression. In *Proceedings of the international joint conference on artificial intelligence (IJCAI*, pages 1208–1214. IJCAI, 2015.

[Blei *et al.*, 2003] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

[Breiman, 2001] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[Bridle, 1990] John S Bridle. Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. In *Neurocomputing*, pages 227–236. Springer, 1990.

[Deshpande *et al.*, 2007] Pawan Deshpande, Regina Barzilay, and David R Karger. Randomized decoding for selection-and-ordering problems. In *HLT-NAACL*, pages 444–451, 2007.

[Řehůřek *et al.*, 2011] Radim Řehůřek, Petr Sojka, et al. Gensim—statistical semantics in python. 2011.

[Filippova, 2010] Katja Filippova. Multi-sentence compression: finding shortest paths in word graphs. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 322–330. Association for Computational Linguistics, 2010.

[Ganitkevitch *et al.*, 2013] Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. Ppdb: The paraphrase database. In *HLT-NAACL*, pages 758–764, 2013.

[Kohlschütter *et al.*, 2010] Christian Kohlschütter, Peter Fankhauser, and Wolfgang Nejdl. Boilerplate detection using shallow text features. In *Proceedings of the third ACM international conference on Web search and data mining*, pages 441–450. ACM, 2010.

[Lapata and Barzilay, 2005] Mirella Lapata and Regina Barzilay. Automatic evaluation of text coherence: Models and representations. In *IJCAI*, volume 5, pages 1085–1090, 2005.

[Lapata, 2003] Mirella Lapata. Probabilistic text structuring: Experiments with sentence ordering. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 545–552. Association for Computational Linguistics, 2003.

[Le and Mikolov, 2014] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1188–1196, 2014.

[Lin, 2004] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 74–81, 2004.

[Nishikawa *et al.*, 2010] Hitoshi Nishikawa, Takaaki Hasegawa, Yoshihiro Matsuo, and Genichiro Kikui. Opinion summarization with integer linear programming formulation for sentence extraction and ordering. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 910–918. Association for Computational Linguistics, 2010.

[Parveen and Strube, 2015] Daraksha Parveen and Michael Strube. Integrating importance, non-redundancy and coherence in graph-based extractive summarization. In *Proceedings of the 24th International Conference on Artificial Intelligence*, pages 1298–1304. AAAI Press, 2015.

[Riloff and Lorenzen, 1999] Ellen Riloff and Jeffrey Lorenzen. Extraction-based text categorization: Generating domain-specific role relationships automatically. In *Natural language information retrieval*, pages 167–196. Springer, 1999.

[Sauper and Barzilay, 2009] Christina Sauper and Regina Barzilay. Automatically generating wikipedia articles: A structure-aware approach. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 208–216. Association for Computational Linguistics, 2009.

[Zhao and Karypis, 2002] Ying Zhao and George Karypis. Evaluation of hierarchical clustering algorithms for document datasets. In *Proceedings of the eleventh international conference on Information and knowledge management*, pages 515–524. ACM, 2002.