DTATG: An Automatic Title Generator Based on Dependency Trees

Liqun Shao and Jie Wang
Department of Computer Science, University of Massachusetts, One University Avenue, 01854, Lowell, MA, U.S.A.
shaoliqun89@gmail.com, wang@cs.uml.edu

Keywords: Title Generator, Central Sentence, Sentence Compression, Dependency Tree Pruning, TF-IDF

Abstract: We study automatic title generation for a given block of text and present a method called DTATG to generate titles. DTATG first extracts a small number of central sentences that convey the main meanings of the text and are in a suitable structure for conversion into a title. DTATG then constructs a dependency tree for each of these sentences and removes certain branches using a Dependency Tree Compression Model we devise. We also devise a title test to determine if a sentence can be used as a title. If a trimmed sentence passes the title test, then it becomes a title candidate. DTATG selects the title candidate with the highest ranking score as the final title. Our experiments showed that DTATG can generate adequate titles. We also showed that DTATG-generated titles have higher F1 scores than those generated by the previous methods.

1 INTRODUCTION

An adequate title for a given block of text must convey succinctly the central meanings of the text. A good title must also be catchy. Writers would typically go through multiple rounds of revisions to come up with a satisfactory title. Automatic title generation (ATG) for a given block of text aims to generate a title comparable to a title composed by humans. For simplicity, we will simply call a block of text a document.

We approach ATG in two phases. In the first phase we search for a central sentence that is "close" to being a title in the following sense: (1) It captures the central meanings of the text; (2) It is in a form that can be converted into a title without too much effort. We treat such a sentence as a title candidate. In the second phase we compress this title candidate into the final title. More specifically, during the first phase, we extract a few central sentences that capture the main meanings of the document. We do so using keyword-extraction algorithms to extract keywords and rank these sentences based on the number of keywords it contains. During the second phase, we construct a set of rules to select a central sentence in a suitable form. We then construct a dependency tree for each central sentence using a dependency parser. We trim possible branches according to a set of empirical rules we devise, and to form the final title. We name our system Dependency-Tree Automatic Title Generator (DTATG).

If a document that already has a title, and we are asked to generate an alternative title for the document, then we can augment the above algorithm by generating title candidates in the first phase as follows: After computing central sentences, we further compute the similarities of these sentences to the exiting title, and select the ones with higher similarities.

ATG may also be used in applications where we are asked to compose an article on a given title. We may use ATG to generate a title of what we have written and compute the similarity of the generated title to the given title. The article would be considered on the right track if the two titles are sufficiently similar. In applications where we may need to paraphrase the original document, we can also use ATG to generate a new title, which may be more suitable to the new document.

We may also use ATG to generate a label for a topic in a document, where a topic is represented by a set of words relevant to that topic. For example, given a document in a corpus, we may first use the LDA algorithm (David M. Blei and Jordan, 2003) to identify the topics contained in the document, where each topic is a set of words, and then use ATG to generate a short phrase or a short sentence to label each set.

Keyword extractions and sentence compressions are basic ATG building blocks. Keywords provide a compact representation of the content of a document, where a keyword is a sequence of one or more words. Hence, a sentence having a larger number of keywords is expected to better convey the main meanings of a document. A popular method to identify key-

words is to use the TF-IDF measure. Given a document in a corpus, a term in the document with a higher TF-IDF value implies that it appears more frequently in the document, and less frequently in the remaining documents. However, for a corpus of a small number of documents, the TF-IDF value of a keyword would almost equal to its frequency.

The *Word Co-occurrence* (WCO) method (Matsuo and Ishizuka, 2004) is a better method, which applies to a single document without a corpus. WCO first extracts frequent terms from the document, and then collects a set of of word pairs co-occurring in the same sentences (sentences include titles and subtitles), where one of the words is a frequent term. If term t co-occurs frequently with a subset of frequent terms, then it is likely to be a keyword. The authors of WCO showed that WCO offers comparable performance to TF-IDF without the presence of a corpus.

The *Rapid Automatic Keyword Extraction* (RAKE) algorithm (Rose et al., 2010) is another keyword extraction method based on word pair co-occurrences. In particular, RAKE first divides the document into words and phrases using predetermined word delimiters, phrase delimiters, and positions of stop words. RAKE then computes a weighted graph, where each word is a node, and a pair of words are connected with weight n if they co-occur n times. RAKE then assigns a score to each keyword candidate, which is the summation of scores of words contained in the keyword. Word scores may be calculated by word frequency, word degree, or ratio of degree to frequency. Keyword candidates with top scores are then selected as keywords for the document. RAKE is superior over WCO in the following aspects: It is simpler and achieves a higher precision rate and about the same recall rate. We will use RAKE to extract keywords.

Early title generation methods include Naive Bayesian with limited vocabulary, Naive Bayesian with full vocabulary, Term frequency and inverse document frequency (TF-IDF), K-nearest neighbor, and Iterative Expectation Maximization. These methods, however, only generate an unordered set of keywords as a title without concerning syntax. Using the F1 score metric as a base for comparisons, it was shown through extensive experiments that the TF-IDF title generation method has the best performance over the other five methods (Jin and Hauptmann, 2001).

For practical purposes we would like to generate syntactically correct titles. Recent methods used sentence trimming to convert a title candidate into a shorter sentence or phrase, while trying to maintain syntactic correctness. Sentence trimming has been studied in recent years and has met with certain success. For example, Knight and Marcu (Knight and Marcu, 2002) and Turner and Charniak (Turner and Charniak, 2005) used a language model (e.g., the trigram model) to trim sentences. Vandegehinste and Pan (Vandegehinste and Pan, 2004) used context-free grammar (CFG) trees to trim a sentence to generate subtitle candidates with appropriate pronunciations for the deaf and hard-of-hearing people. They used a spoken Dutch corpus for evaluation. More recently, Zhang et al. (Zhang et al., 2013) used sentence trimming on the Chinese text to generate titles, also through CFG trees. We note that CFG is ambiguous and the complexity of constructing CFG trees is high.

Dependency trees are recent development in natural language processing with a number of advantages. For example, dependency trees offer better syntactic representations of sentences (Sylvain, 2012) and they are easier to work with. These advantages motivated us to explore automatic title generation using dependency trees. We present the first such algorithm.

Our approach has the following major differences from the previous approaches:

1. We use RAKE to generate keywords and define a better measure to select central sentences.
2. We use dependency grammar to construct a dependency tree for each title candidate for trimming.
3. We construct a set of empirical rules to generate titles.

We show that, through experiments, DTATG generates titles comparable to titles generated by human writers. In addition to this evaluation, we also evaluate the F1 scores and show that, through experiments, DTATG is superior over the TF-IDF method (see Section 4.3).

The remainder of this paper is organized as follows: In Section 2 we first provide an overview of DTATG. We then explain DTATG in detail, including extraction of central sentences, dependency parsing, and the dependency tree compression model. In Section 4 we describe our experiment evaluation setups and present results from our experiments. We conclude the paper in Section 5.

2 DETAIL DESCRIPTION OF DTATG

The DTATG system framework to generate a title for a given document is shown in Figure 1, where we assume that each document already has a title for comparison. If a document does not have a title, then this comparison will not be executed. Given a document, DTATG generates a title as follows:

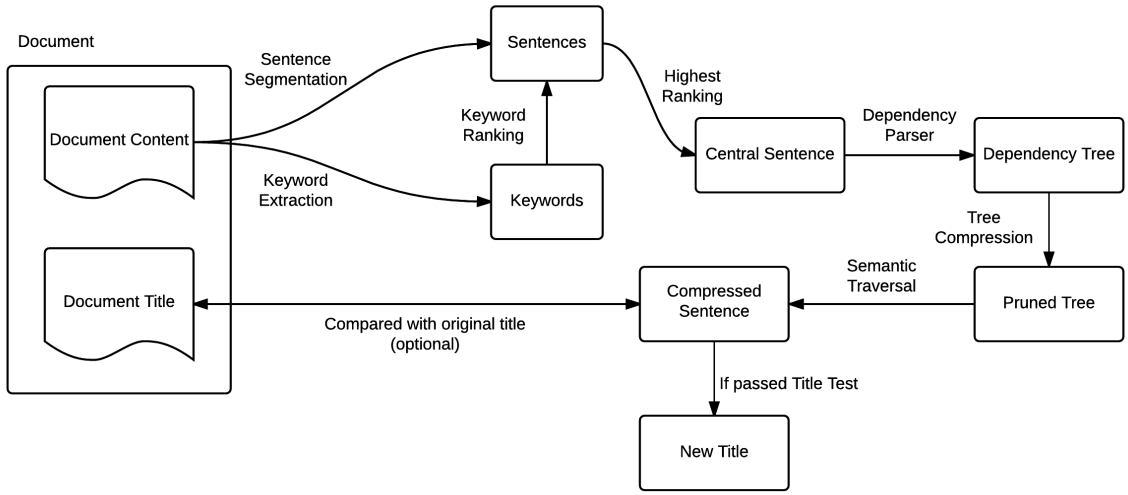

Figure 1: DTATG system framework.

1. Extract keywords.
2. Use sentence segmentation to obtain sentences and rank each sentence using a suitable measure based on the number of keywords it contains. Select a fixed number of sentences with the highest rankings as the central sentences. In general, selecting three central sentences would be sufficient.
3. Construct a dependency tree for each central sentence using a dependency parser, starting from the sentence with the highest ranking. In particular, we use Stanford University's open-source tool—Stanford Dependency Parser—as our dependency parser.
4. Remove certain branches of the dependency tree based on a set of empirical rules we devise to compress the sentence.
5. If the trimmed sentence passes the title test we devise, then output it as the title.

However, we note that not all documents have central sentences. For example, in a document that describes a number of items to be avoided, it may simply state that "The following items should be avoided:" followed by a list of items. In this document, any sentence that describes an item does not include the keyword "avoided", and the sentence that contains this keyword does not contain any keywords for describing any item. Thus, none of the sentence in the document can represent the central meaning of the document, that is, none of the sentences can specify what items should be avoided.

For convenience, we call documents with central sentences **type-1 documents** and documents without central sentences **type-2 documents**. While DTATG may apply to both types of documents, it performs better for type-1 documents.

2.1 Central sentence extraction

We segment each document into sentences using a sentence-delimiter set {., ?, !, \n, :}. This set does not include comma as compared to sentence delimiters used by other researchers, for we want to obtain a complete sentence. From experience we should select only consider sentences with at most 25 words for title candidates.

We first use RAKE (Rose et al., 2010) to identify keywords, where RAKE assigns each keyword with a positive numerical score. Next we will compute the ranking of each sentence to reflect the importance of the sentence based on the keywords it contains. Assume that a sentence contains n keywords $w_1, \cdots, w_n$, and w_i has a positive score s_i. An obvious method to define the rank of the sentence is to simply sum up the scores of all the keywords contained in it as follows:

$$\text{Rank}_1 = \sum_{i=1}^{n} s_i. \quad (1)$$

However, we observe that using Rank_1 we may end up giving a sentence containing a larger number of keywords of lower scores a higher rank than a sentence containing fewer keywords of higher scores. For example, suppose that sentence S_1 contains two keywords with scores of $(4,5)$ and sentence S_2 contains four keywords with scores of $(2,2,3,3)$. Then S_2 has a higher Rank_1 ranking than S_1. This contradicts to the common sense that S_1 would be more relevant to the central meanings.

Thus, we would want keywords of higher scores to carry more weights, so that the ranking of a sentence with smaller number of keywords of much higher scores is higher than the ranking of a sentence with

larger number of keywords of much lower scores. To achieve this, we use the power of 2 to amplify the scores, giving rise to the following measure:

$$\text{Rank}_2 = \sum_{i=1}^{n} 2^{s_i}. \quad (2)$$

In the example above, it is easy to see that S_1 has a higher Rank_2 ranking. On the other hand, if S_2 contains three keywords each with a score of 4, then we would consider S_1 and S_2 equally important. The Rank_2 measure ensures this effect. Moreover, if S_2 contains four keywords with scores of $(1,4,4,4)$, then we would want S_2 to have a higher ranking than S_1. Again, the Rank_2 measure also ensures this effect.

Empirical results indicate that Rank_2 is a better measure for ranking sentences. For example, Table 1 shows two central sentences obtained using Rank_1 and Rank_2 from a news article as an example, both having the highest score under the respective measure. It is evident that, compared with the original title, the central sentence obtained using Rank_2 presents a better choice.

Table 1: Examples of central sentence extraction

Original title	Bad e-mail habits sustains spam
Rank_1	People must resist their basic instincts to buy from spam mails
Rank_2	One in ten users have bought products advertised in junk mail

DTATG uses Rank_2 to select central sentences as follows: Order sentences first by rank and then by the sentence length. In other words, we first select the sentence with the highest rank and if two sentences have the highest rank, we select the shorter one. Our experiments indicate that this method of selecting central sentences improves performance.

The sentence we selected may have multiple clauses separated by commas. If the sentence has more than two clauses, we keep the two clauses that have the highest ranking and separate them using a space. This makes it easier to perform sentence compression in the next step.

To generate suitable sentences for trimming, we apply the following empirical rules to avoid inappropriate sentences.

- Discard sentences that contain a pronoun subject.
- Discard sentences that contain predicate verbs in {am, is, are}.

2.2 Dependency trees

We use the Stanford Dependency Parser (SDP), available at `http://nlp.stanford.edu/software/stanford-dependencies.shtml`, to obtain grammatical relations between words in a sentence. The relations are presented in triplets:

(relation, governor, dependent).

The tree is constructed by the relations between each word in the sentence. For example, *nsubj* stands for nominal subject, which is the syntactic subject of a clause. Universal Dependencies are documented online at `http://universaldependencies.org/u/dep/index.html`. For example, Figure 2 depicts a dependency tree for the sentence "Microsoft warned PC users to update their systems". In addition, we may use the Stanford Parser (SP), available at `http://nlp.stanford.edu/software/lex-parser.shtml`, to obtain part-of-speech tagging.

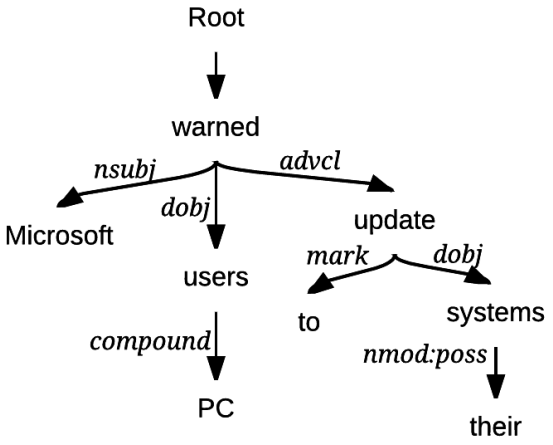

Figure 2: The dependency tree of sentence "Microsoft warned PC users to update their systems"

2.3 Dependency tree compression model

We devise a dependency tree compress model (DTCM) to generate titles. First, we define the following set of empirical rules to specify what cannot be trimmed. These rules were formed based on our experiences from working with a large number of text documents. We refer to these rules as the **to-be-kept** rules.

1. If the relation is *nsubj* or *nsubjpass*, then keep both the governor and dependent.
2. If the relation is *dobj* or *iobj*, then keep both the governor and dependent.
3. If the relation is *compound*, then keep both the governor and dependent.
4. If the relation is *root*, then keep the dependent.

5. If the relation is *nmod*, then keep the dependent and the preposition in *nmod.*
6. If the relation is *nummod*, then keep both the governor and dependent.

We note that the remainder of the sentence after removing all the other relations not specified the to-be-kept rules can still be understandable by humans.

DTCM proceeds as follows:

1. Traverse the entire dependency tree in preorder.
2. For each leaf node t, if it is not a keyword or a part of the keyword or the relation of its parent edge is not in the set of to-be-kept rules, then remove node t.

In general, DTCM deletes all the unnecessary branches; most of these branches would have relations such as *det*, *aux*, and *amod*.

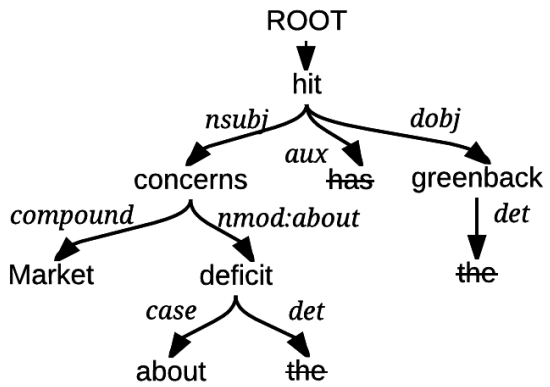

Figure 3: The trimmed dependency tree using DTCM for sentence "Market concerns about the deficit has hit the greenback."

Figure 3 shows a trimmed dependency tree using DTCM for a central sentence "Market concerns about the deficit has hit the greenback." The output is "Market concerns about deficit hit greenback", which would be a suitable title.

2.4 Compression rate

To transform a trimmed dependency tree into a title, we output the words in the same order of the original sentence. Let y be a compressed sentence for a sentence $x = w_1 w_2 w_3 \cdots w_n$, where w_i is a word. We use "0" to mark a word to be removed and "1" to mark a word to be retained. Then a compression rate r for a sentence is defined as follows:

$$r = \frac{\text{\# of 1s in } y}{n}. \quad (3)$$

If r is too large or too small, we may delete more or fewer words, respectively, to make the output more reasonable. For example, since the length of a good title should not exceed 10 words (this is a commonsense rule), if a central sentence is more than 20 words and the compression rate r is less than 50% after pruning, then we may apply DTCM again on the trimmed sentence with the hope to obtain a higher compression rate.

The following is a set of empirical rules for deletion and we refer to them as **to-be-deleted** rules. We may keep adding rules to this set.

- Delete the first adverbial phrase and the last adverbial phrase.
- Delete "X says" and "X said", where X is a noun or pronoun.
- If there are two clauses connected with "and" and the first clause has subject and verb, delete the second clause with "and".
- If there is a clause starting with "that" and the clause has a subject and a verb, then delete all the words before "that" (including "that").
- If there are more than one *nmod* relations next to each other, then delete all the *nmod* relations except the last one.

3 TITLE TEST

A good title must pass the **title test**, which consists of the following three individual tests.

The conciseness test.

- A title should not exceed 15 words.
- A title must not have clauses.
- A title should have the following structure: "Subject + Verb + Object" or "Subject + Verb". Subject must be specific: it can be a noun but not a pronoun.

The fluency test. A title should contain no grammatical errors.

The topic-relevance test. A title must convey at least one major meaning of the document.

DTATG uses central sentences to achieve topic relevance, and uses dependency grammar and empirical rules to achieve conciseness and fluency. We will use the title test to evaluate the quality of the generated titles by DTATG.

4 EVALUATIONS

We evaluate DTATG on a corpus of English documents obtained from the BBC news website (Greene and Cunningham, 2006). The dataset can be found at `http://mlg.ucd.ie/datasets/bbc.html` and `https://drive.google.com/open?`

id=0B_-6yYneQ1_8Ml91TnY0THJoYXc, which consists of 2,225 documents in five topical areas from the year of 2004 to the year of 2005. These five topical areas are business, entertainment, politics, sport, and tech. The corpus provides raw text files with original articles and titles. The datasets include type-1 and type-2 documents. In our experiments, we will use type-1 articles in the business, entertainment, politics, and tech categories and type-2 articles in the sport category.

4.1 Experiment setup

We carried out experiments on a computer with 2.7 GHz dual-core Intel Core i5 CPU and 8 GB memory. In our experiments, we used a Python implementation of the RAKE algorithm to extract keywords, and the Stanford Parser to generate dependency trees, where the source code for implementing RAKE can be found at https://github.com/aneesha/RAKE and the Stanford Parser can be downloaded from http://nlp.stanford.edu/software/lex-parser.shtml#Download.

4.2 Results and discussions

We randomly selected 50 articles in each category to evaluate. We asked English speakers to rank each DTATG-generated title in the category of conciseness, fluency, and topic relevance using a 5-point scale.

Rating definition on conciseness (CON)

5: Excellent; every word in the sentence is to the point that makes a perfect title.
4: Very good; the sentence contains no redundant words that makes a very good title.
3: Good; the sentence has some redundant words, but may still be used as a title.
2: Weak; the sentence has some redundant words, but is a poor choice as a title.
1: Poor; the sentence contains a lot of redundant words and should not be used as a title at all.

Rating definition on fluency (FLU)

5: Excellent; the sentences contains no grammatical flaws.
4: Very good; the sentences contains only minor grammatical flaws.
3: Good; the sentence contains some grammatical flaws.
2: Weak; the sentence is only partially intelligible, with serious grammatical flaws.
1: Poor; the sentence makes no sense at all.

Rating definition on topic relevance (TR)

5: Excellent; the sentence has perfect topic relevancy.
4: Very good; the sentence has significant topic relevancy.
3: Good; the sentence has moderate topic relevancy.
2: Weak; the sentence has marginal topic relevancy.
1: Poor; the sentence is topic irrelevant.

Evaluation results

Each evaluation category was evaluated by 5 human evaluators. We then average their scores. On the conciseness and fluency evaluation, we asked the evaluators to compare the original titles with DTATG-generated titles. On the topic relevance evaluation, we asked the evaluators to make a subjective judgment on how much the DTATG-generated titles represent the main topics of the article. For this evaluation, the evaluators needed to read the original articles. We note that most articles in the sport category are type-2 documents. In these documents we simply used the first sentence as the central sentence.

To better assess the quality of the DTATG results, we also ask evaluators to evaluate the original titles using the same rules, and we use the original as the comparison baseline. Results are given in Table 2 and Figure 4 for a better visual. In Figure 4, the dotted lines represent the scores of titles generated by DTATG and the solid lines represent the scores of original titles.

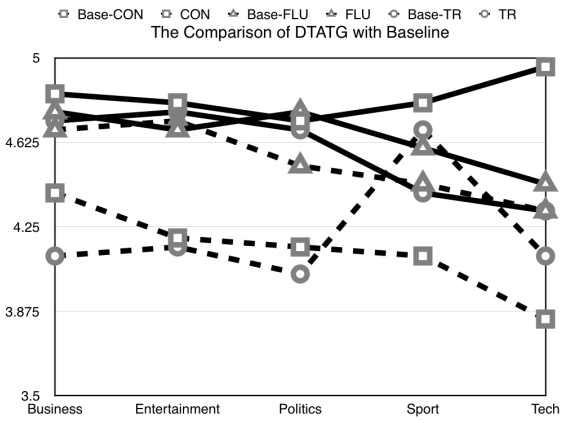

Figure 4: Line graph of DTATG-generated titles and original titles on CON, FLU, and TR

From Table 2 we can see that the titles generated by DTATG have scores over 4 on fluency and topic relevance on all categories. Moreover, most of the titles are concise except in the tech category. The reason why the generated titles for the tech articles

Table 2: The average scores for DTATG-generated titles compared with the original titles

Category	Base-CON	CON	Base-FLU	FLU	Base-TR	TR
Business	4.84	4.40	4.76	4.68	4.72	4.12
Entertainment	4.80	4.20	4.68	4.72	4.76	4.16
Politics	4.72	4.16	4.76	4.52	4.68	4.04
Sport	4.80	4.12	4.60	4.44	4.40	4.68
Tech	4.96	3.84	4.44	4.32	4.32	4.12

Table 3: Comparison examples of DTATG-generated titles with the original title showing the score (CON, FLU, TR)

Original title	Blogs take on the mainstream (5, 5, 4)
Central sentence	The blogging movement has been building up for many years.
DTATG title	Blogging movement building up for years (5, 5, 5)
Remark	The DTATG title contains more information, and so we consider it better than the original title.
Original title	Day-Lewis set for Berlin honour (5, 5, 4)
Central sentence	Japan's oldest film studio will also be honoured along with Day-Lewis.
DTATG title	Japan's oldest film studio honoured with Day-Lewis (5, 5, 5)
Remark	The original article talked about both Day-Lewis's and Japan's oldest film studio's contributions. The original title only mentioned Day-Lewis. The DTATG title mentioned both, and so we would consider it a better title.
Original title	Scots smoking ban details set out (5, 5, 5)
Central sentence	A comprehensive ban on smoking in all enclosed public places in Scotland.
DTATG title	Comprehensive ban on smoking in public places in Scotland (5, 5, 5)
Remark	The DTATG title is as good as the original title made by a professional writer.
Original title	Xbox 2 may be unveiled in summer (5, 5, 5)
Central sentence	Since its launch, Microsoft has sold 19.9 million units worldwide.
DTATG title	Microsoft sold 19.9 million units worldwide (5, 5, 4)
Remark	The DTATG title is missing Xbox as the name of the units sold, and so it is not a very good title.

are not as concise is because these articles often contain long technology names or long company names, which make titles longer. Nevertheless, DTATG in general does reasonably well on all evaluation categories of conciseness, fluency, and topic relevance. DTATG-generated titles are understandable and acceptable by human evaluators. From Figure 4, we can easily tell that titles generated by DTATG are close to baselines. Titles of FLU in Entertainment and TR in Sport categories are better than the baseline, which means they are even better than the original titles because they provide more information (see Table 3).

Detailed information of these examples, including the original title, the DTATG-generated title, the original text, the keywords, the central sentence, the dependency tree for the central sentence, and the pruned branches, can be found at `http://119.29.118.23/title_generator/index.html`. We also implemented a system at `http://www.cwant.cn/title_en/` that allows users to enter text, upload a txt file, or enter an URL to generate a title automatically using DTATG.

We note that the performance of DTATG is confined by the choice of central sentences. To produce a good title, we must have a good central sentence to begin with. For example, in the last example in Table 3, if "Xbox" is included in the central sentence to read "Since its launch, Microsoft has sold 19.9 million Xbox units worldwide", then the DTATG-generated title would have been "Microsoft sold 19.9 million Xbox units worldwide", which would be a good title. However, the original article mainly talked about the success of Xbox and only mentioned briefly that Xbox 2 would be available in May. Thus, no central sentences would be able to capture Xbox 2.

4.3 Comparision with TF-IDF

For title generation using the TF-IDF method, the words in the document with highest TF-IDF would be chosen for the title. Given a document, let T1 denote the title generated by DTATG and T2 the original title. Let precision denote the number of identical words in

T1 and T2, divided by the number of words in T1; and recall the number of identical words in T1 and T2, divided the number of words in T2. Then the F1 score is defined by

$$\text{F1} = \frac{2 \times precision \times recall}{precision + recall}. \quad (4)$$

We randomly selected 100 articles from datasets in Section 4 and computed the F1 score for each article. The average F1 scores are shown in Figure 5.

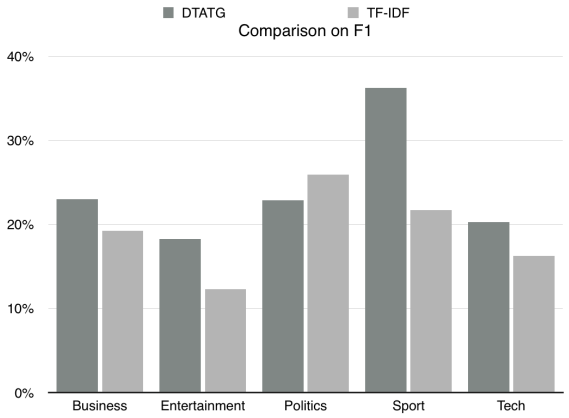

Figure 5: Comparison of DTATG and TF-IDF on the F1 score

In four out of five categories, the titles generated by DTATG perform better than those generated by TF-IDF. In particular, for the category of sport, the average F1 score under DTATG is much higher than that under TF-IDF. In the category of politics, the average F1 score under DTATG is slightly lower than that under TF-IDF.

5 CONCLUSION

We presented DTATG for automatic title generation on a given block of text. DTATG is a system combining central sentence selection and dependency tree pruning. DTATG is unsupervised and can generate titles quickly and syntactically. DTATG, however, is confined by the choice of central sentences. We plan to study how to combine several central sentences and compress them using deletion, substitution, reordering, and insertion to generated better titles.

ACKNOWLEDGEMENTS

This work was supported in part by a research grant from Wantology LLC. We thank Yiqi Bai, Shan (Ivory) Lu, Jing Ni, Jingwen (Jessica) Wang, Hao Zhang, and a few other people for helping us evaluate the titles generated by DTATG.

REFERENCES

David M. Blei, A. Y. N. and Jordan, M. I. (2003). Latent dirichlet allocation. *The Journal of machine Learning research*, 3(2003): 993–1022.

D. Greene and P. Cunningham. (2006). Practical solutions to the problem of diagonal dominance in kernel document clustering. In *ICML*.

Kevin Knight and Daniel Marcu. (2002). Summarization beyond sentence extraction: A probabilistic approach to sentence compression. In *Artificial Intelligence*, 139(1): 91-107.

Rong Jin and Alexander G. Hauptmann. (2001). Automatic Title Generation for Spoken Broadcast News. In *Proceedings of HLT-01*, 2001, pp. 1ĺC3.

Stuart Rose, Dave Engel, Nick Cramer and Wendy Cowley. (2010). Automatic keyword extraction from individual documents. In *Text mining applications and theory*, 2010, pp. 3-19.

Jenine Turner and Eugene Charniak. (2005). Supervised and unsupervised learning for sentence compression. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, Ann Arbor, Mich., 25-30 June 2005, pp. 290-297.

Vincent Vandegehinste and Yi Pan. (2004). Sentence compression for automated subtitling: A hybrid approach. In *Proceedings of the ACL-04*, 2004 (pp. 89-95).

Y. Matsuo and M. Ishizuka. (2004). Keyword extraction from a single document using word co-occurrence statistical information. *International Journal on Artificial Intelligence Tools*, 13(1): 157-169.

Sylvain Kahane. (2012). Why to choose dependency rather than constituency for syntax: a formal point of view. In *Meanings, Texts, and other exciting things: A Festschrift to Commemorate the 80th Anniversary of Professor Igor A. Mel'čuk*, Languages of Slavic Culture, Moscou, pp. 257-272.

Yonglei Zhang, Cheng Peng, and Hongling Wang. (2013). Research on Chinese Sentence Compression for the Title Generation. *Chinese Lexical Semantics*, edited by Xiao Guozheng & Ji Donhong. Heideberg: Springer.