Postprint: Uyghur Text Similarity Detection Using N-gram and Semantic Analysis

Authors: Zhang Ying, Yasin Eziz, Wu Shunxiang

Date: 2018-05-24T00:00:00+00:00

Abstract

Currently, most natural language text similarity estimation approaches are primarily designed for major languages such as English. To enable similarity detection for Uyghur texts, this paper proposes a similarity detection method based on N-gram and semantic analysis. First, according to the characteristics of Uyghur words, an N-gram statistical model is employed to extract words, and a word-text relationship matrix is constructed based on word frequencies in the text to serve as the text model. Then, Latent Semantic Analysis (LSA) is utilized to capture the latent associations between words and texts, thereby addressing the issue of semantic ambiguity in Uyghur and obtaining accurate similarity measures. Experiments conducted on a plagiarism corpus containing text reordering and synonym replacement demonstrate that the proposed method can accurately and effectively detect similarities.

Full Text

Preamble

Uyghur Text Similarity Detection Method Using N-gram and Semantic Analysis

Zhang Ying[1], Yasen Aizezi[1]†, Wu Shunxiang[2] [1](Dept. of Information Security Engineering, Xinjiang Police College, Urumqi Xinjiang 830013, China) [2](Dept. of Automation, Xiamen University, Xiamen Fujian 361005, China)

Abstract: Currently, most natural language text similarity estimation research focuses on major languages such as English. To enable similarity detection for Uyghur texts, this paper proposes a similarity detection method based on N-gram and semantic analysis. First, according to Uyghur word characteristics, the N-gram statistical model is employed to obtain words, and a word-text relationship matrix is constructed based on word frequency in texts as the text

model. Then, Latent Semantic Analysis (LSA) is adopted to obtain hidden associations between words and their texts, thereby addressing the problem of ambiguous word meanings in Uyghur and obtaining accurate similarity measures. Experiments on plagiarized text sets containing reorganization and synonym replacement demonstrate that the proposed method can detect similarities accurately and effectively.

Key words: Uyghur language; text similarity detection; N-gram statistical model; latent semantic analysis

0 Introduction

With easy access to information via the internet, academic plagiarism has become a simple operation. Several types of document plagiarism exist: (a) directly copying phrases or paragraphs from published texts without providing citations or attributing the author; and (b) modifying statements and structures of published content for use. To protect authors' copyrights, plagiarism detection for published documents represents an important measure. Some similarity estimation and plagiarism detection methods are language-independent and applicable to multiple languages, while others are language-sensitive. Language-independent methods estimate similarity based on text features that are not inherent to specific natural languages, such as character counts and average sentence length values. Language-sensitive methods, based on attributes specific to individual languages, offer higher relevance and accuracy than language-independent approaches.

In recent years, with the economic and educational development of Xinjiang, numerous academic papers written in Uyghur have emerged. Conducting similarity computation and plagiarism detection for Uyghur documents holds significant importance for the healthy development of Uyghur culture. This paper addresses Uyghur text similarity detection. Due to potential morphological variations, synonyms, and multiple meanings in Uyghur words, each word may have different forms, with prefixes and suffixes attachable to words in continuous patterns. A single string may contain verb inflections, prepositional variations, pronoun variations, and conjunction variations. Consequently, semantic comparison of Uyghur text words becomes ambiguous, posing certain difficulties for plagiarism detection.

Since the development of Uyghur document information processing began relatively late, current research on Uyghur document similarity primarily originates from Xinjiang University. Due to the complex linguistic structure of Uyghur, some commonly used similarity measures cannot be well applied. Few scholars have proposed relevant methods to date. Literature [7] proposes a Uyghur sentence similarity calculation method (MUSM) that employs morphological features and calculates similarity between two Uyghur sentences through a multi-strategy selection algorithm. However, it can only perform detection at the sentence level and does not consider synonym replacement issues. Literature

[8] first introduces and analyzes Uyghur text semantic similarity measurement, determining semantic similarity through context to address synonym problems, but its accuracy is relatively low.

This paper proposes a Uyghur text similarity detection method based on N-gram and semantic analysis. Its main innovations are: (a) employing the N-gram statistical model to obtain word stems according to Uyghur word characteristics, and constructing a text model based on word frequency; and (b) adopting Latent Semantic Analysis (LSA) to obtain hidden associations between words and texts, thereby addressing the problem of ambiguous word meanings in Uyghur and obtaining accurate similarity measures. Experimental results demonstrate that the proposed method can accurately and effectively detect plagiarized texts containing reorganization and synonym replacement.

1 Basic Framework of the Proposed Method

[Figure 1: see original paper] Framework of the proposed text similarity analysis method

The proposed method primarily consists of several components: text preprocessing, word extraction, text modeling, and similarity analysis. The preprocessing stage includes text normalization, tokenization, and useless word removal operations. The word extraction stage mainly involves word extraction and sorting using N-gram technology. The text modeling stage includes TF-IDF matrix calculation and word matching. The similarity stage includes Singular Value Decomposition (SVD) and latent semantic analysis.

1.1 Uyghur Language Features

Uyghur is a script based on Arabic letters with high agglutination. The Uyghur alphabet contains 32 letters with diverse forms, typically including four representation forms, resulting in relatively complex morphological changes. Uyghur words consist of stems and affixes, with different affixes added before and after the same stem to represent different meanings. Due to these features, Uyghur text information processing encounters certain difficulties, such as high feature dimensionality.

Table 1 demonstrates words formed by adding different affixes before and after the stem “ ” (author) and their meanings, where affixes are underlined.

Table 1 Words formed by adding affixes to the stem “ ” (author)

Word Form	Affixes	Meaning
	-	author (female)
	-	the author
	-	like the author
	-	similar to the author

Word Form	Affixes	Meaning
	-	of the author (female)
	-	belonging to the author
	-	becoming an author
	-	my author

1.2 Basic Framework

The objective of this paper is to develop a similarity analysis method for natural language texts. The proposed method can operate in two modes. The first mode analyzes similarity between texts, including suspicious and reference texts. The second mode involves a single input as a text-based query, with the output being similarity measures between texts or queries.

In the proposed method, we assume that original and rewritten texts both contain measurable differences that can be captured through statistical and linguistic indicators. To overcome difficulties in similarity/plagiarism detection for Uyghur texts, this paper employs three main technical approaches. The first adopts Natural Language Processing (NLP) techniques rather than relying on traditional string matching methods. The second employs text modeling techniques capable of overcoming substantial lexical and syntactic challenges. The third uses Latent Semantic Analysis (LSA) to determine hidden associations contained within texts, making it a viable technique for natural language text plagiarism detection.

During preprocessing, the indexing module reads texts sequentially, generating word indexes for each statement and passing these indexes to the N-gram counting module. The N-gram counting module writes N-gram words generated from each text into separate temporary files. These temporary files are merged into a single file, and N-gram structures are sorted to remove duplicate counts. Subsequently, the text modeling module reads the sorted N-gram structures to compute the TF-IDF matrix, which serves as the feature matrix for the given text set. The similarity estimation module then calculates cosine similarity between texts through the feature matrix as a preliminary similarity estimate. The TF-IDF matrix is subsequently passed to the LSA function for deep estimation of similarity within the text set. The main data processing flowchart is shown in Figure 2.

[Figure 2: see original paper] Data processing flowchart for similarity estimation

2 Text Similarity Estimation Steps

2.1 Text Preprocessing

Text preprocessing represents an important prerequisite for successful natural language processing tasks. First, the input text set is converted to plain text,

with all control characters filtered out. Next, texts are parsed for Part-of-Speech (PoS) tagging, employing the Uyghur language model in this work.

For each text, each statement is tokenized and stored in memory. Figure 3 shows a text tokenization example for the Uyghur text “ ” (translation: This water is really good). Word indexes are obtained by invoking a morphological analyzer to acquire part-of-speech analysis for each inflected word. These analyses are used for disambiguation, applying relevant PoS tags to each inflected word. PoS tagging can resolve potential morphological ambiguity in words. If multiple possible part-of-speech words still exist, the Levenshtein edit distance is used to select the most probable stem, choosing the inflected word with the minimum edit distance to the possible stem. This work employs the morphological analyzer and Uyghur lexical query adopted in literature [12]. This morphological analyzer is developed based on linguistic methods, using stem indexes stored in a dictionary to index inflected words according to the selected stem.

[Figure 3: see original paper] Example of text tokenization

Stop words are removed during indexing by examining the morphological features of each inflected word. If the feature values indicate that the current word is an interjection, preposition, or pronoun, the inflected word is considered a stop word.

2.2 Word Extraction Based on N-gram Statistical Model

In applications such as text classification and detection, words must first be extracted from texts. This paper adopts statistical methods more suitable for the Uyghur environment to extract words. The employed statistical method is the N-gram statistical model, which segments words at the letter level by treating consecutive N letters as a gram unit.

In the N-gram model, for a specific letter in text, its occurrence probability is assumed to be related to the preceding N-1 letters. Therefore, the probability of a letter sequence occurring is calculated as:

$$P(l_1, l_2, ..., l_N) = \prod_{i=1}^{N} P(l_i | l_1, ..., l_{i-1})$$

The N value in the N-gram model must be determined in combination with the specific language environment. For Uyghur, since each word consists of multiple letters combined together, a smaller N cannot effectively represent word attributes, while a larger N (such as 3 or 4) exhibits stronger representational capability.

In the word extraction process using the N-gram statistical model, to reduce word dimensionality and redundancy, we first remove the most common affixes in words according to the Uyghur dictionary. Then, we calculate the similarity between two words to extract word stems.

To demonstrate the word extraction process using the N-gram statistical model, an example with N=2 is presented, calculating the similarity between the two words “ ” (revolution) and “ ” (revolutionary). First, the words are decomposed into N=2 letter combination units: “ ” “ ” → “ ”, “ ”, “ ”. After removing common affix two-letter combinations → “ ”, “ ”. The similarity between these two words is calculated as:

$$CS = \frac{2 \times C}{A + B + 2 \times C}$$

where A represents the number of letter combinations present in the first word but not in the second word; B represents the number of letter combinations present in the second word but not in the first word; and C represents the number of identical letter combinations present in both words. If the similarity between two words exceeds a set threshold, the two words are merged into one stem.

The N-gram counting steps must be continuous for each gram size. For example, for unigram, bigram, and trigram, the program must run three times. To avoid massive storage requirements, N-grams are extracted from individual data blocks, and only one N-gram size is kept in memory at a time to maximize efficiency. The basic process of N-gram counting steps is shown in Figure 4.

[Figure 4: see original paper] Data processing flowchart for N-gram counting

The most effective method for plagiarism detection uses Document Frequency (DF) as a feature. To reduce the number of words in texts, we determine whether some words are important based on their document frequency. Words that exist in only one text are immediately removed because they cannot be plagiarized in any other text. Additionally, we remove words contained in more than $\mu+\sigma$ texts (where μ is the average document frequency and σ is the standard deviation of document frequency). In other words, all common words are removed from the text.

2.3 Constructing the Text Model

This paper proposes a word-text model considering word frequency in texts. The relationship between these words and texts is represented in matrix form, where columns represent texts and rows represent words. Consider m vectors $A_1, A_2, ..., A_m$ composed of n vectors, where vector A_j represents words contained in text j.

Each vector A_j consists of n elements a_{ij}, where a_{ij} represents the weight of word i frequency in text j, as shown in equation (2). This equation is the weight coefficient proposed in this paper for constructing the feature matrix A, representing a modified version of standard TF-IDF weighting.

$$a_{ij} = \begin{cases} \frac{\log(PF_{ij})}{\log(DF_i)} \cdot \frac{1}{\sqrt{\max(PF_{ij})}}, & \text{if } PF_{ij} \in [2, N] \\ \log(DF_i) \cdot \log(M) \cdot 0.5, & \text{if } 1 < i < j \\ 0, & \text{otherwise} \end{cases}$$

where PF_{ij} represents the frequency of word i appearing in text j, DF_i represents the number of texts containing word i, and M is the total number of all texts. Compared with TF-IDF, the difference in the proposed frequency weight calculation method lies in the normalization of IDF. We divide it by $\log(\max(PF_{ij}))$. On the other hand, if word i does not appear in text j, then $a_{ij} = 0$. This weighting mechanism helps the subsequently adopted SVD achieve optimal results.

During TF-IDF matrix A construction, pairwise N-gram word matching is performed. This is a direct comparison process with overall complexity of $O(N^2)$, where N is the number of individual words in the considered texts. When considering lexical and syntactic variations, the complexity of the entire pairing process increases to $O(N^3)$. For such cases, some techniques can be used to estimate pairwise word matching scores. In this work, we use matching average and Dice coefficient in this matching process. This is performed by representing the relationship between each pair of tokenized words in matrix form to calculate matching scores.

The matrix "cost" calculation method for pairwise word matching is: if the index of token i in the first word equals the index of token j in the second word and its synonyms/antonyms, then $cost_{ij} = 1$; otherwise, $cost_{ij} = 0$. The matching score value indicates whether the considered pair of words is identical. In this work, if the matching score equals 1.00, the pair of words is considered identical.

2.4 Latent Semantic Analysis

This stage infers potential semantic associations between words contained in texts. LSA is an intelligent text comparison technique that uses mathematical algorithms to analyze large amounts of text and reveal underlying semantic information, making it a viable technique for natural language text plagiarism detection.

This paper uses a linear algebra technique that diagonalizes symmetric matrices: Singular Value Decomposition (SVD), which decomposes matrix A into three independent matrices: left singular matrix U, singular matrix Σ, and right singular matrix V. Matrix Σ contains only diagonal elements called singular values, while matrices U and V contain detailed decomposition information.

All these matrices can be decomposed in the latent space k to perform the optimal k-level approximation of A, where singular values $\sigma_{k+1}, \sigma_{k+2}, ..., \sigma_r$ are replaced with 0, with $r \leq \min(m, n)$ and r being the number of non-zero diagonal elements in Σ. Then, matrix U is an $m \times k$ column orthogonal matrix whose

columns are word singular vectors. Σ is a $k \times k$ diagonal matrix containing k largest singular values in A. Matrix V^T is a $k \times n$ orthogonal matrix whose rows are text singular vectors.

A characteristic of SVD is that singular values on the diagonal of Σ are arranged in descending order, satisfying equation (3). Matrix V^T is an $n \times n$ orthogonal matrix whose rows are text singular vectors.

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_k \geq \sigma_{k+1} \geq \cdots \geq \sigma_r > 0$$

Figure 5 presents the SVD decomposition process. The matrix A_k obtained after decomposition serves as the basic building block for further processing because it contains independent profile vectors of texts.

[Figure 5: see original paper] Singular Value Decomposition of the text-word matrix

The approximation error between the original word-text matrix A and A_k is determined by the truncated (or discarded) singular values $(\sigma_{k+1}, \sigma_{k+2}, ..., \sigma_r)$. The relative change reflected by approximating A through A_k is estimated by $\frac{\|A-A_k\|_F}{\|A\|_F}$, where the Frobenius matrix norm $(\|\cdot\|_F)$ for real numbers is defined as:

$$\|B\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} b_{ij}^2}$$

By setting all singular values in Σ except the k largest values to zero, we can approximately construct A_k with rank k. The approximation error between A and A_k is $\|A - A_k\|_F = \sqrt{\sum_{i=k+1}^{r} \sigma_i^2}$.

2.5 Similarity Estimation

The final stage calculates pairwise text similarity. Before using matrix B, we must rescale all individual elements of text profiles with corresponding singular values as follows:

$$B = U_k \Sigma_k V_k^T$$

Then, correlation is calculated according to equation (5), where column lengths of matrix B are normalized. The resulting sim_{SVD} is a symmetric matrix where each pair of texts is represented by a score indicating similarity.

$$sim_{SVD} = B^T B$$

Reducing words helps texts achieve higher sim_{SVD} scores, as the vast majority of phrases are meaningless and need to be deleted. Therefore, this paper modifies the matrix sim_{SVD} calculation formula as shown in equation (6). The resulting estimate represents the overall situation of corresponding similarity measurement.

$$sim_{SVD} = \begin{cases} sim(R,S) \cdot \frac{N(R_{red})}{N(R_{orig})}, & \text{if } \frac{N(R_{red})}{N(R_{orig})} \cdot \frac{N(S_{red})}{N(S_{orig})} < \tau \\ sim(R,S), & \text{otherwise} \end{cases}$$

where τ is the similarity threshold.

If the query vector q is calculated using weighted frequencies corresponding to the specified type in the query, it represents a suspicious text compared with columns of matrix A_k (corresponding to the original word-text matrix A). Assuming vector e_j represents the j-th canonical vector of dimension m (i.e., the j-th column of the identity matrix I_m), then vector q is the j-th column of matrix A_k with rank k. For text vector a_j, the cosine value of the angle between query vector q and the k-dimensional text vector (or column) b_j of A_k can be expressed by the following formula:

$$\cos(\theta_j) = \frac{(U_k^T q)^T (U_k^T a_j)}{\|U_k^T q\| \cdot \|U_k^T a_j\|}, \quad j = 1, 2, ..., n$$

3 Experiments

3.1 Experimental Setup

To evaluate the proposed method's performance in estimating similarity between Uyghur texts and potential textual plagiarism (including word reorganization and synonym replacement), a dataset containing 9 Uyghur texts (L1~L9) is used for testing. The texts contain specific quantities of useless stop words and useful words, as shown in Figure 6.

[Figure 6: see original paper] Nine texts for similarity estimation

To construct plagiarized texts containing reorganization and synonym replacement, the first 5 texts in the dataset are designated as original texts. Text 6 is extracted from Text 3. Text 7 consists of two parts: one from Text 3 and another from Text 4. Text 8 is an exact copy of Text 7, but 50% of words are changed to their synonyms. The final text is generated from Text 7 but with 50% of statements reorganized. Figure 7 shows the actual similarity relationships among the 9 considered texts.

[Figure 7: see original paper] Actual similarity among the nine texts

All experiments are conducted on an Intel Core i7-4700 CPU with a 2.4 GHz clock speed and Microsoft Windows 8 system platform. The proposed algorithm is implemented through MATLAB compilation for similarity estimation.

3.2 Parameter Selection

To obtain the optimal N value in the N-gram algorithm, we set its value between 1 and 6 to measure the precision, recall, and F-measure values of similarity estimation. The plagiarism text similarity threshold τ is set to 30%, meaning that when similarity between two texts reaches 30%, it is considered plagiarism.

To obtain statistically meaningful comparison results, the algorithm' s detection performance is repeated 30 times across 9 texts. The average results are shown in Figure 8. It can be observed that detection performance varies under different N values. Performance is not ideal when N is either too large or too small, but relatively superior results can be achieved when N=3 or 4. This is because when N is small, the obtained words cannot sufficiently express real meanings. When N is large, it increases the semantic dimensionality of the approximate matrix A_k, negatively impacting similarity measurement estimation.

[Figure 8: see original paper] Performance metrics of the proposed method under different N values

To more accurately demonstrate performance differences, referencing real similarity values, we statistically analyze the maximum and average absolute differences between the estimated pairwise text similarity and real values across 30 experiments of the proposed method, as shown in Table 2. The conclusion can be drawn that results obtained using N=3 in N-gram outperform other results.

Table 2 Maximum and average differences between estimated and real similarity values

N-gram Value	Maximum	Difference
1	28.74%	12.82%
2	17.66%	3.32%
3	12.27%	2.25%
4	13.62%	2.57%
5	21.72%	3.54%
6	25.87%	4.71%

Additionally, as the N value increases, the computation time of the proposed method also increases, as shown in Figure 9. Therefore, considering both detection performance and computation time comprehensively, N=3 is ultimately selected.

[Figure 9: see original paper] Computation time of the proposed method under different N values

3.3 Performance Comparison

The proposed detection method is compared with the similarity detection method proposed in literature [8], where N=3 is set in our method. Experiments

are conducted on the aforementioned 9 texts, and similarity estimation and detection performance results are shown in Table 3.

Table 3 Performance comparison results

Metric	Literature [8] Method	Proposed Method
Maximum	Difference	in similarity estimation
Average	Difference	in similarity estimation
Precision	92.8%	99.4%
Recall	85.6%	88.6%
F-measure	92.5%	95.5%

The results show that the proposed method outperforms the literature [8] method because our approach is based on natural language processing and well-suited for complex languages like Uyghur. The literature [8] method includes stop words in similarity estimation, resulting in higher similarity estimates.

Generally, in the absence of synonym replacement, both methods produce similar similarity estimation results, particularly for reorganized text sets. However, in synonym replacement scenarios, the proposed method performs better than the literature [8] method when measuring similarity across all synthetic plagiarized texts.

Therefore, comprehensive experimental results demonstrate that the proposed method can accurately estimate similarity of Uyghur texts with morphological variations, reorganization, and synonym replacement.

4 Conclusion

This paper proposes a method specifically for Uyghur text similarity estimation. Based on the relationship model between texts and N-gram words, PoS tagging is applied to examined texts to support resolution of morphological ambiguity during text normalization. Through text indexing and stop word removal, a text TF-IDF model is constructed. Finally, LSA and SVD are used to investigate hidden associations between texts and words. Experimental results prove that the proposed method demonstrates strong capability in detecting text similarity, capable of addressing plagiarism involving copying, sentence reordering, and synonym replacement.

The similarity detection method proposed in this paper is designed for Uyghur texts, employing the N-gram statistical model to obtain word stems—a step unnecessary for English, Chinese, and other languages. Additionally, the method of obtaining hidden associations between words and texts based on LSA can be applied to other languages, helping improve detection of ambiguous language texts.

In future work, we will attempt more effective word matching methods to improve efficiency in detecting statement variations. Additionally, parallel algorithms will be considered for processing large-scale texts.

References

[1] Zou Du, Chen Yuqing, Zhang Ling. A plagiarism detection method based on semantic matching [J]. Journal of South China University of Technology: Natural Science Edition, 2013, 41 (7): 131-136.

[2] Zhang Chao, Chen Li, Li Qiong. A Chinese text similarity calculation method based on PST_LDA [J]. Application Research of Computers, 2016, 33 (2): 375-377.

[3] Barron-Cedeno A, Gupta P, Rosso P. Methods for cross-language plagiarism detection [J]. Knowledge-Based Systems, 2013, 50 (1): 211-217.

[4] Turdi Tohti, Winira Musajan, Askar Hamdulla. Semantic string-based topic similarity measuring approach for Uyghur classification [J]. Journal of Chinese Information Processing, 2017, 31 (4): 100-107.

[5] Sindhu L, Idicula Sumam Mary. A plagiarism detection system for Malayalam text based documents with full and partial copy [J]. Procedia Technology, 2016, 25 (4): 372-377.

[6] Maimaitiyiming Hasimu, Wushouer Silamu, Weinila Mushajiang, et al. Research N-gram based Uyghur text classification technique [J]. Application Research of Computers, 2015, 32 (7): 1986-1988.

[7] Tian Shengwei, Turgun Ibrahim, Yu Long, et al. Similarity measure algorithm of Uyghur sentence [J]. Computer Engineering and Applications, 2009, 45 (26): 144-146.

[8] Ma Bo, Zhou Xi, Yang Yating, et al. Uyghur semantic similarity computation based on contextual information in web documents [J]. Journal of Computational Information Systems, 2012, 8 (2): 563-570.

[9] Yan Chenggang, Xie Hongtao, Liu Shun, et al. Effective Uyghur language text detection in complex background images for traffic prompt identification [J]. IEEE Trans on Intelligent Transportation Systems, 2017, 19 (1): 1-10.

[10] Mi Chenggang, Yang Yating, Wang Lei, et al. Detection of loan words in Uyghur texts [J]. Communications in Computer & Information Science, 2014, 49 (6): 103-112.

[11] Jiang Zongli, Wang Wei. Translation recommendation system with information retrieval technology [J]. Journal of Harbin Engineering University, 2017, 38 (3): 419-424.

[12] Boudchiche M, Mazroui A, Bebah M O A O, et al. AlKhalil Morpho Sys II: a robust Arabic morpho-syntactic analyzer [C]// Proc of International Conference

on Information Technology for Organizations Development. 2016: 23-28.

[13] Zhang Xinwei, Wu Bin. Short text classification based on feature extension using the N-gram model [C]// Proc of International Conference on Fuzzy Systems and Knowledge Discovery. Piscataway, NJ: IEEE Press, 2016: 710-714.

[14] Liu Dexi, Nie Jianyun, Zhang Jing, et al. Extracting sentimental lexicons from Chinese microblog: a classification method using N-gram features [J]. Journal of Chinese Information Processing, 2016, 30 (4): 193-205.

[15] He Tianwen, Wang Hong. Evaluating perplexity of Chinese sentences based on grammar & semantics analysis [J]. Application Research of Computers, 2017, 34 (12): 3538-3542.

Note: Figure translations are in progress. See original paper for figures.