algorithms

Article

Tsetlin Machine for Sentiment Analysis and Spam Review Detection in Chinese

Xuanyu Zhang [1], Hao Zhou [1], Ke Yu [1,*], Xiaofei Wu [1] and Anis Yazidi [2]

1 School of Artificial Intelligence, Beijing University of Posts and Telecommunications, Beijing 100876, China
2 Department of Computer Science, Oslo Metropolitan University, 0176 Oslo, Norway
* Correspondence: yuke@bupt.edu.cn

Abstract: In Natural Language Processing (NLP), deep-learning neural networks have superior performance but pose transparency and explainability barriers, due to their black box nature, and, thus, there is lack of trustworthiness. On the other hand, classical machine learning techniques are intuitive and easy to understand but often cannot perform satisfactorily. Fortunately, many research studies have recently indicated that the newly introduced model, Tsetlin Machine (TM), has reliable performance and, at the same time, enjoys human-level interpretability by nature, which is a promising approach to trade off effectiveness and interpretability. However, nearly all of the related works so far have concentrated on the English language, while research on other languages is relatively scarce. So, we propose a novel method, based on the TM model, in which the learning process is transparent and easily-understandable for Chinese NLP tasks. Our model can learn semantic information in the Chinese language by clauses. For evaluation, we conducted experiments in two domains, namely sentiment analysis and spam review detection. The experimental results showed thatm for both domains, our method could provide higher accuracy and a higher F1 score than complex, but non-transparent, deep-learning models, such as BERT and ERINE.

Keywords: tsetlin machine; interpretability analysis; spam review detection; Chinese sentiment analysis

check for updates

Citation: Zhang, X.; Zhou, H.; Yu, K.; Wu, X.; Yazidi, A. Tsetlin Machine for Sentiment Analysis and Spam Review Detection in Chinese. *Algorithms* **2023**, *16*, 93. https://doi.org/10.3390/a16020093

Academic Editor: Fabio Massimo Zanzotto

Received: 20 December 2022
Revised: 30 January 2023
Accepted: 1 February 2023
Published: 8 February 2023

1. Introduction

Shallow machine learning models may have intuitive interpretability, but they suffer from a performance gap compared with their counterpart deep learning models. With the introduction of BERT [1], pre-training and fine-tuning methods were shown to be excellent in nearly all NLP tasks. These deep learning models achieved superior performance in Chinese NLP tasks, but their black-box nature makes it difficult to explain decisions, which may limit their applicability in certain scenarios.

In this article, we address those challenges by proposing a novel interpretable framework based on the Tsetlin machine (TM), which was originally proposed by [2]. TM is capable of decomposing the problem into multiple pattern recognition problems [2]. Firstly, the binary representations of texts are obtained as the input features for TM. Further, multiple conjunction clauses are applied to learn the implicit propositional logic of the problem. Finally, the output of clauses are aggregated and the classification result is obtained through a thresholding mechanism. Differing from the current widely-used deep learning models, TM has a relatively simple and universal learning mechanism, namely Tsetlin Automata (TA). The learning and feedback mechanism of TA makes TM efficient in terms of resource consumption [3], and yields faster convergence rate [4,5], better interpretability [6] and higher reliability than deep learning models.

Extensive experiments are conducted on three public Chinese datasets, ChnSentiCorp-htl-all, Weibo-Senti and Douban Movie Reviews, for sentiment analysis and spam review detection. The results indicated that our method achieved superior accuracy and a superior

F1 score, compared to state-of-the-art deep learning based methods, e.g., BERT and ERINE [7]. This suggests that TM is a promising tool in NLPm due to its superior performance and excellent interpretability. (A preliminary and very abridged version of some of these results was presented in [8] at the First International Symposium on the Tsetlin (ISTM 2022), in June, 2022, in Grimstad, Norway.)

Our contributions include the following points:

- We use the clauses in the TM to learn the state transition of Chinese words and to obtain optimal states through the feedback mechanism. Once we find out the mapping relationship between words and target classes, we summarize the results of each clause and capture the overall semantic information. To the best of our knowledge, this is the first work that utilizes TM in Chinese NLP tasks.
- We propose a new framework, based on TM, for spam review detection. Compared to previous work on TM, we add richer input features, including user features and product features, etc., and adopt a conversion mechanism to convert these into the form required for model input. To the best of our knowledge, this is the first work that utilizes TM in spam review detection tasks.
- Our devised framework provides a good F1 score and accuracy close to, or higher than, BERT [1] on some open datasets.
- In the case study experiment, the visualization of the decision-making process in our model is demonstrated, which provides an intuitive and easy way to interpret the final output of the proposed model.

The rest of the paper is organized as follows. We list some related works in Section 2. Section 3 describes the processing operation and model structure, and we also introduce the learning and feedback mechanism of the TM model. We present the experimental results and provide some interpretability analysis in Section 4. In Section 5, we draw the concluding remarks of our work.

2. Related Work

Sentiment analysis is a fundamental NLP task and admits a wide range of real-life applications. Shallow machine learning models, such as Support Vector Machine (SVM), Logistic Regression, Decision Trees and Naive Bayes [9], were widely explored in early research. With the rise of deep learning technologies, sentence expression models, such as RNN, CNN [10] and their variant versions, were proposed [11–15]. In addition, in order to better integrate or extract more features, attention mechanisms [16] and models [17] based on Transformer also emerged.

With the accelerated evolution of the internet, more and more users tend to share their experiences and opinions on review sites. The content of review sites cover movies, hotels, and products to events. However, not all reviews are genuine and credible, as many comments are spam, written to gain profits or fame or to undermine the reputation of a product with fake reviews. These spam reviews might mislead consumers about the target product and might negatively impact user experiences. Spam review detection denotes a new NLP field that aims to combat this phenomenon. Spam review detection is mainly based on compound features of words, emotion and user behavior. Before the pre-training model was proposed, people tended to use N-Gram features to represent the meaning of comment text. For example, Fusilier D H et al. [18] proposed a method using character n-grams as features to capture lexical content, as well as stylistic information, in spam opinion. Cagnina and Rosso [19] proposed using character-level n-gram features in reviews and support vector machines as classifiers. Lau et al. [20] combined text mining and semantic analysis to design deceptive review detection methods, which was shown to yield good performance in practical applications. Ott et al. [21] utilized psychological analysis and computational linguistic features to analyze spam comments, and proposed three detection methods, which achieved promising experimental results. Yuan et al. [22] devised multiple attention units and fusion attention units to capture the relationship between the reviewer and the product in order to detect spam comments. Wu et al. [23]

firstly detected spammers with graph neural network, then employed a Markov random field to correct the classification results and improve accuracy.

Although the above-mentioned approaches are extensively utilized, they have their shortcomings. The aforementioned machine learning models use manually engineered features as inputs, which may have poor performance for complex problems. On the other hand, despite the fact that significant advances have been achieved in deep learning, their black-box nature makes them less trustworthy.

Actually, TM has achieved excellent results in many NLP works. Yadav [24] resorted to TM to convert the input of aspect-based sentiment analysis into a binary form of limited information, introduced additional knowledge from sentiwordnet, and built an interpretative model with TM. Bhattarai et al. [25] used TM's conjunction clauses to learn and represent known classes and established a novelty scoring mechanism. In a subsequent work, Bhattarai et al. [26] adopted TM clauses to capture the vocabulary and semantic attributes of true or false news to predict news authenticity, and to calculate the credibility of fake news. Berge et al. [27] applied TM on the IMDb data set as well as on a medical data set. They converted complex text content into simple propositional formulae with good interpretability. Saha et al. [28] resorted to the TM model to mine the relationship between emotions and semantics.

In this article, we used TM to perform Chinese NLP tasks, including sentiment analysis and spam review detection. We did not only extract comment word features, but also we used additional features to represent user behavior and product attributes. Extensive experiments were conducted on two public datasets, and the results demonstrated that our proposed framework achieved improved performance, while enjoying high interpretability.

3. TM for Chinese NLP Tasks

3.1. Architecture Overview

We divided the overall model into three parts according to their functions, namely Input Module, Clause Module and Output Module. In the Input Module, we used bag of words (*BOW*) to binarize Chinese texts into one-hot vector form. In the spam review detection task, following previous works, the features of users and products were introduced as additional features, which could help the model to detect deliberately fabricated reviews. This was equivalent to adding prior knowledge to the model. Specifically, we converted the user and target features into a binary form and then text features and additional features were fused together as the final output of the module (See Section 3.2 for details).

As Figure 1 shows, the Clause Module consisted of a series of clauses which evaluated the input vector. The clauses were composed of a series of Tsetlin Automata (TA) teams, where TA represents not negated literal, and TA' negated literal. They decide whether the literal is excluded or included in the clause, as in Equation (1) shown (see Section 3.3 for details and Section 3.5 for learning process).

$$\begin{gathered} Clause_i^w = f(TA_1, {TA_1}', ..., TA_n, {TA_n}'), \\ i \in \{1, 2, ..., c\}, w \in \{0, 1\}, \end{gathered} \tag{1}$$

where $Clause_i^w$ is the clause, and function $f(\cdot)$ is affected by the action of TA and TA'. w is the polarity of Clause, and c is the number of Clauses, which can be configured manually. Finally, each Clause of the TA Module outputs a score for each category. In the Output Module, all scores are summarized, and the highest one considered as the final output category.

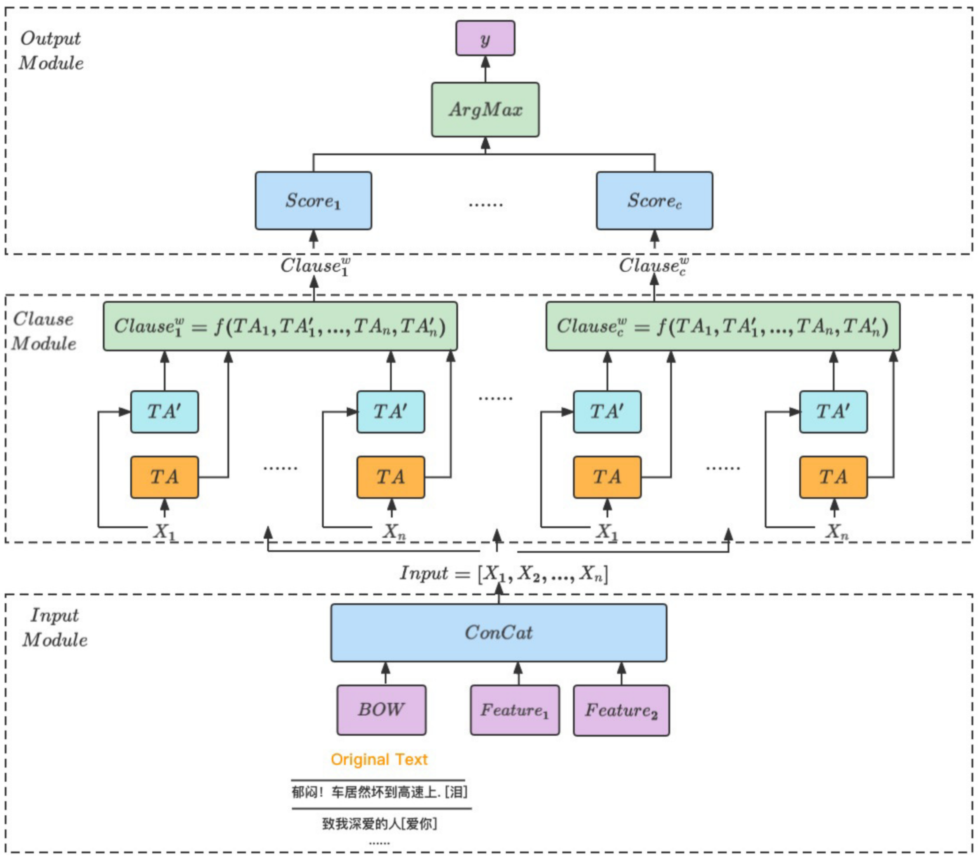


Figure 1. The Architecture of TM. In the Input Module, BOW means one-hot vector from bag of words and Feature means external features. We used a dotted line to frame the Feature extraction process in the Input Module to indicate that this step was optional. In the TA Module, we calculated the output of clause by function $f(\cdot)$. Parameter c and *class* were set manually. One was the clauses number and the other was the classes number. Function $Score_i$ sums all outputs of clauses for the target class.

3.2. Input Module

The input module provides the Chinese texts' embedding vectors for post modules. TM requires input features to be binary, so that it can decompose the problem into multiple sub-patterns.

We binarized reviews into one-hot form. As *BOW* might lead to an overly sparse distribution problem, we referred to a previous work [29] to alleviate this. Firstly, we calculated the cosine similarity of word embeddings. Then, we treated words with similar meanings as a whole and marked them together. On the other hand, we kept only high-frequency words and used chi-square test to identify the importance of each word and selected the key words. Bag size was reduced 3 times without missing much useful information. For the spam review detection task, we introduced additional features, including user behavior and product-related features. Additional features could increase the prior knowledge of the model. The TM model learning process automatically learnt the relationship between these features and categories and, finally, improved performance. The statistical features of the users' behaviors and products' properties were used in our framework, such as the mean, maximum and minimum values of the voting numbers, the length of related comments and the percentage of spam reviews. We adopted product features associated with the comments, which were actually movie features from Douban, an influential online Chinese platform for reviews of movies, music and books.

These additional features also needed to be converted to the binary input form required by TM, so we adopted the following conversion mechanism:

1. First, for each feature, the unique values were identified and then sorted from smallest to largest;

2. The sorted unique values were considered as thresholds and saved in the feature matrix U;

3. We compared the original features with their own identified thresholds. We set 1 or 0 depending on whether the feature value was greater than the threshold.

4. We repeated the above three steps until we achieved the final Boolean vector.

For additional features, including movie feature and user feature, we first calculated different values of each column in the feature matrix. For convenience, we only retained no more than 25 different values for each feature and saved them in the Matrix U. Each row in U represented one additional input feature, and the width of that row was the amount of different values of that feature (no more than 25 in our experiment). For example, binary conversion could be performed according to the following rules:

$$Output[:, pos] = (RawVector[:, i] >= U[i][j]) \tag{2}$$

In the Equation (2), i represents the index of a certain feature from input $RawVector$ (one column of X), j represents the index of that feature's different values saved before in U (j is smaller than 25 in our experiment).

For the sake of illustration, we considered one sample's additional feature $RawVector[1, d]$ as an example (d represents the dimension of the feature). As shown in Figure 2, if one dimension of the $RawVector$ was smaller than another of the different values $U[i][j]$, previously counted in U (where j is the index of different values of feature i, j smaller than 25), then the corresponding output was marked as 0, and otherwise marked as 1. In this way, any feature of input was expanded to the amount of different values (maximum 25).

This was equivalent to a statistical judgment of the relationship between every element of a feature and all elements of that feature.

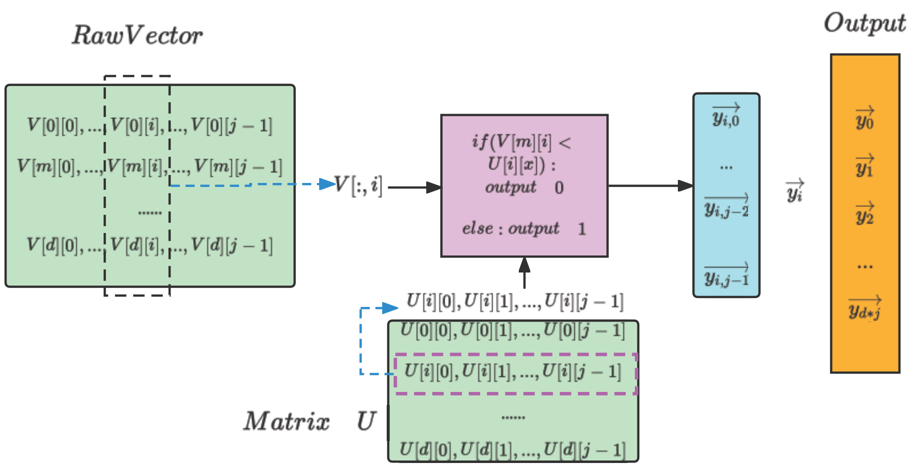

$RawVector$
$V[0][0], ..., V[0][i], ..., V[0][j-1]$
$V[m][0], ..., V[m][i], ..., V[m][j-1]$
......
$V[d][0], ..., V[d][i], ..., V[d][j-1]$
$V[:, i]$
$if(V[m][i] < U[i][x]):$
$output \quad 0$
$else: output \quad 1$
$U[i][0], U[i][1], ..., U[i][j-1]$
$U[0][0], U[0][1], ..., U[0][j-1]$
$U[i][0], U[i][1], ..., U[i][j-1]$
......
$U[d][0], U[d][1], ..., U[d][j-1]$
$Matrix \quad U$
$\overrightarrow{y_{i,0}}$
...
$\overrightarrow{y_{i,j-2}}$
$\overrightarrow{y_{i,j-1}}$
$\overrightarrow{y_i}$
$Output$
$\overrightarrow{y_0}$
$\overrightarrow{y_1}$
$\overrightarrow{y_2}$
...
$\overrightarrow{y_{d*j}}$

Figure 2. Transform Mechanism. *RawVector* was the input and y was the output. We assumed the size of input was d, and i represented the value of ith feature of input. U was a matrix containing each feature's different values, its shape was $[d, j]$ (j was not fixed in different rows but was definitely smaller than 25).

3.3. Clause Module

The clause module was composed of a series of Tsetlin Automata (TA), which could automatically learn the best action in an unknown random environment so as to learn a set of sub-patterns expressed in propositional logic. That meant that a feature, or in other words, a sub-pattern, was captured by a group of TAs, which formed a clause. One TA was only responsible for including or excluding a literal within a clause. The complex relationship was, therefore, captured by distinct clauses jointly for a certain class.

As shown in the Figure 1, the sub-patterns associated with the target classes were captured by conjunction clauses. The number of clauses namely c was manually set. We assumed that the dimension of feature formulated at the input module was n. To be clear, n was not fixed, it was determined by the size of the word bag on the current condition. For each feature X_i, we set two-action TAs, namely TA and TA', to learn the original state and the opposite state of X_i. The opposite one was called $\neg X_i$, as Equation (3) shows:

$$\neg X_i = 1 - X_i, i \in \{1, 2, \ldots, n\}, X_i \in \{0, 1\} \tag{3}$$

As Figure 3 shows each TA or TA' had N states and performed Include or Exclude action based on the current state. Then it triggered a reward or penalty. Include Action represented the inclusion of current literal, and Exclude Action meant the abandonment of the current literal. TA learnt the state change of input literal X_i, and TA' learnt the change of the opposite input literal $\neg X_i$. Function $f(\cdot)$ calculated the output of all TA and TA', namely TA_{index} and $TA^{'}_{index}$, and formed Clause $Clause^w_i$, where index was the index of dimensions, i was the index of Clauses and w meant the polarity of Clause ($w = 1$ represented positive and $w = 0$ represented negative). The feedback mechanism affected the choice of each action, based on whether the action was reasonable, by rewarding or punishing. When a TA received a reward, it emphasized the action performed by moving towards the most internal state. However, if a penalty happened, the TA moved towards the center to reduce the reliance on the current action and weaken it, eventually switching to the other action.

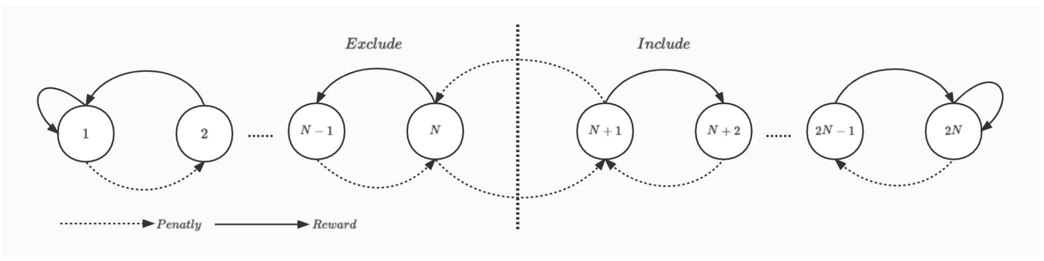


Figure 3. Two Actions and Transition of TA.

TM used multiple clauses to enhance generalization ability and output a better judgment result. Half the clauses learned positive polarity, and the other half learned negative polarity. For convenience, we used odd indices ($i = 1, 3, 5 \ldots$) for clauses with positive polarity $Clause^1_i$, and even indices ($i = 2, 4, 6 \ldots$) for clauses with negative polarity $Clause^0_i$.

As an example, the current literal was k, and the clause's index was i, and the polarity was w. If i was an odd number, the current clause learnt positive aspects ($w = 1$). The calculation formula of $Clause^1_i$ was given by Equation (4), where $\wedge$ meant multiply and c was the number of clauses (set manually):

$$\begin{aligned} Clause^1_i &= f(TA_1, TA^{'}_1, \ldots, TA_n, TA^{'}_n) \\ &= TA_1 \wedge TA^{'}_1 \wedge \ldots \wedge TA_n \wedge TA^{'}_n, i \in \{1, \ldots, c\}. \end{aligned} \tag{4}$$

On the contrary, if i was an even number, the current clause learnt a negative aspect ($w = 0$). The calculation of $Clause^0_i$ is given by Equation (5):

$$\begin{aligned} Clause^0_i &= f(TA_1, TA^{'}_1, \ldots, TA_n, TA^{'}_n) \\ &= TA_1 \wedge TA^{'}_1 \wedge \ldots \wedge TA_n \wedge TA^{'}_n, i \in \{1, ..., c\}. \end{aligned} \tag{5}$$

3.4. Output Module

We summed the clause outputs of each class as $Score_{class}$. A higher $Score_{class}$ meant a higher chance of the sample in that class. So we assigned the sample into the class that had the highest $Score_{class}$. We calculated the sum of positive terms $Clause_i^1$ at odd positions and deducted the sum of negative terms $Clause_i^0$ at even positions to get the final score of the current category $Score_{class}$, where x was the index of classes:

$$Score_{class} = \sum_{1}^{n} Clause_i^1 - \sum_{1}^{n} Clause_i^0 \tag{6}$$

Finally, we compared all scores and chose the category with the highest score as the final result y. A TM updated parameters online according to every training sample (X, y). A TA team decided the clause output, and collectively, the output of all the clauses decided the TM's output. Hence, it was important to sensibly guide individual TAs in clauses to maximize the accuracy of the TM's output. There were two kinds of reinforcement: Type I and Type II Feedback. One type gave an appropriate feedback to the TA in the clause according to the actual scenario, such as reward or penalty, or inaction feedback. More details are discussed in Section 3.5.

3.5. Interpretative Learning Process

Differing from the neural network's convergence method, that minimizes the output error, learning processing of TM was adjusted according to 4 factors: actual output, prediction output, current state and action of TAs. Considering a particular situation, we analyzed the inclusion of literal in l_k, where j represented the index and w represented the polarity. We defined Inclusion as α_2 and Exclusion as α_1. Once the clause made an Action, a Feedback was triggered.

As shown in Figure 4, when the truth value of Literal was 1 and $Clause_j^w$ was positive polarity (w was 1), $Clause_j^w$ received Reward feedback with a probability of $\frac{s-1}{s}$. s was a hyper parameter (set manually), and it influenced the probability of Reward, Inaction or Penalty. Type I Feedback could reinforce the true positive output of a clause (the clause output was 1 when it had to be 1) by introducing more literals from samples. This made clauses' predictions become closer to 1 (when the label was 1). This helped the model avoid false negatives (the clause output was 0 when it was supposed to be 1). We could also sub-divide the Type I Feedback into Type Ia and Ib. Type Ia impacted the reinforcement of exclude actions while Type Ib reinforced exclude actions of TAs. Together, Type Ia and Type Ib Feedback forced clauses to output 1. To diversify the clauses, they were targeted for Type I Feedback stochastically as follows:

$$p_j = \begin{cases} 1, & with\ probability\ \frac{T-max(-T,min(T,v))}{2T} \\ 0, & otherwise. \end{cases} \tag{7}$$

We wanted different sub-patterns to be learned, which meant it would be better for clauses to be smartly allocated among the sub-patterns. T was a manually set parameter in Equation (7), which meant the number of clauses available to learn each sub-pattern in each class. A higher T allocated more clauses to learn each sub-pattern and increased the robustness of learning. The probability of clause j receiving Type I Feedback, p_j, was influenced by T and v. The decisions for the complete set of clauses to receive Type I Feedback were organized in the vector $P = (p_j)\epsilon\{0,1\}^m$.

Once the clauses to receive Type I Feedback were selected according to Equation (7), the probability of updating individual TA was based on the clause output value, i.e., 1 for Type Ia feedback and 0 for Type Ib feedback. Specifically, the probability of the TA updated according to Type Ia feedback when the literal also had value 1 (the kth TA of the jth clause that received Type Ia Feedback) was given by $r_{j,k}$, shown in Equation (8):

$$r_{j,k} = \begin{cases} 1, & with\ probability\ \frac{s-1}{s} \\ 0, & otherwise. \end{cases} \tag{8}$$

Similarly, the probability for the Type Ib Feedback was given by $q_{j,k}$ in Equation (9)

$$q_{j,k} = \begin{cases} 1, & \textit{with probability } \frac{1}{s} \\ 0, & \textit{otherwise}. \end{cases} \tag{9}$$

For Type Ia, the current state should move more towards the include action direction, while for Type Ib Feedback, the current state should move towards the exclude action direction to strengthen the exclude action of TAs.

In comparison, the mechanism of Type II Feedback in Figure 5 was simpler. It dud not need the adjustment of the hyper parameter s, but made a trade-off with absolute probability. For example, when the true value of Literal was 0 and $Clause_j^w$ had positive polarity, it received a Penalty with a probability of 1. To turn this clause output from 1 to 0, a literal value of 0 could simply be included in the clause. Clauses with a positive polarity needed Type II feedback when $y = 0$ and clauses with negative polarity needed this when $y = 1$ as they did not want to vote for the opposite class. Again, using the user-set target T, the decision for the jth clause was made as follows:

$$p_j = \begin{cases} 1, & \textit{with probability } \frac{T+max(-T,min(T,v))}{2T} \\ 0, & \textit{otherwise} \end{cases} \tag{10}$$

The states of the TAsm having corresponding literals with value 0 in selected clauses were now moved towards the include action direction with a probability of 1, according to Equation (10). Type II Feedback introduced literals that made the clause evaluation false when a false alarm error occurred, which could increase the discrimination of the model and suppress false alarm errors (the clause output 1 when it had to be 0). It made clause predictions closer to zero.

Truth Value of Clause $Clause_j^w$		1		0	
Truth Value of Literal l_k		1	0	1	0
Include Literal ($\alpha_2 \rightarrow l_k \in L_j^w$)	P(Reward)	$\frac{s-1}{s}$	NA	0	0
	P(Inaction)	$\frac{1}{s}$	NA	$\frac{s-1}{s}$	$\frac{s-1}{s}$
	P(Penalty)	0	NA	$\frac{1}{s}$	$\frac{1}{s}$
Exclude Literal ($\alpha_1 \rightarrow l_k$)	P(Reward)	0	$\frac{1}{s}$	$\frac{1}{s}$	$\frac{1}{s}$
	P(Inaction)	$\frac{1}{s}$	$\frac{s-1}{s}$	$\frac{s-1}{s}$	$\frac{s-1}{s}$
	P(Penalty)	$\frac{s-1}{s}$	0	0	0

Figure 4. Type I Feedback.

Truth Value of Clause $Clause_j^w$		1		0	
Truth Value of Literal l_k		1	0	1	0
Include Literal ($\alpha_2 \rightarrow l_k \in L_j^w$)	P(Reward)	0	NA	0	0
	P(Inaction)	1.0	NA	1.0	1.0
	P(Penalty)	0	NA	0	0
Exclude Literal ($\alpha_1 \rightarrow l_k$)	P(Reward)	0	0	0	0
	P(Inaction)	1.0	0	1.0	1.0
	P(Penalty)	0	1.0	0	0

Figure 5. Type II Feedback.

For example, we used an example from a sentiment analysis task to show the learning process of TM: X = 好可爱的名字[可爱], translated into "Such a cute name [cute].", having a positive label y = 1. The input was tokenized and negated: such = 1, cute = 1, name = 1, $\neg such = 0$, $\neg cute = 0$, $\neg name = 0$.

We considered two positive polarity clauses and one negative polarity one (i.e., C_1^+, C_2^+, and C_1^-) to show different feedback conditions occurring while learning the propositional rules. As Figure 6 shows, for the first time stamp t_1 in the figure, we assumed the clauses were initialized randomly, as follows: $C_1^+ = (such \bigwedge cute)$, $C_2^+ = (\neg such \bigwedge name)$ and $C_1^- = (name)$. Since clause C_1^+ consisted of non-negated literals, and output 1, the condition was $C_1^+ = 1$ and $y = 1$. As shown in Equation (8), Type I (a) feedback was used to reinforce the Include action. Therefore, literals, such as *name*, were included for the second time stamp t_2, as Figure 6 shows.

Similarly, for C_2^+, the output was 0 due to $\neg such$. Therefore, Type I (b) feedback was triggered and Excluded actions would start in the condition ($C_2^+ = 0$ and $y = 1$).

Type II feedback (right side of figure) only occurred when y = 0(for positive polarity clauses) and y = 1 (for negative polarity clauses). We took negative polarity clauses C_1^- to demonstrate the effect of Type II feedback. The model added 0-valued inputs to make the output of the affected clause change from 1 to 0.

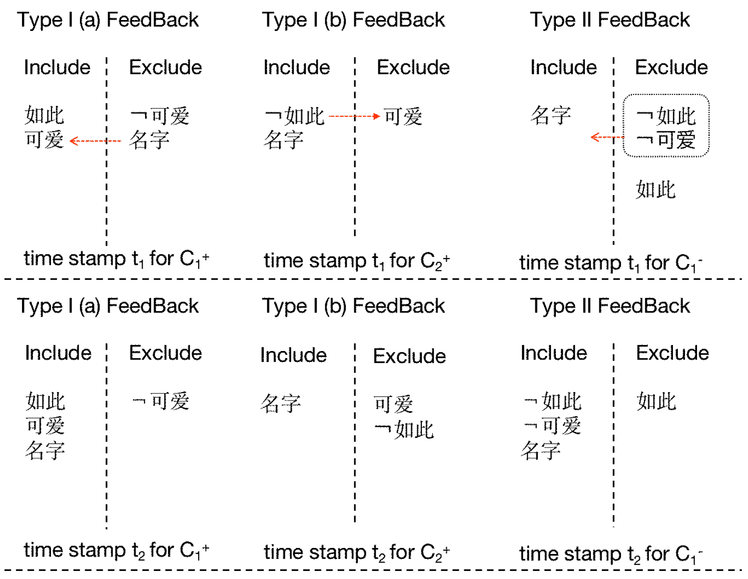


Figure 6. Learning process of Tsetlin Machine.

4. Experiments and Analysis

4.1. Chinese Sentiment Analysis

We chose two public datasets for sentiment analysis, as shown in Table 1. Weibo Dataset had nearly 120,000 samples with two balanced labels. They were collected from Sina Weibo. When it came to ChnSentiCorp-htl-all, it was organized by Songbo Tan, and had 7766 samples with unbalanced distribution of labels. Accuracy and F1 were used as indicators of the model's performance.

Table 1. Stastics of Weibo-Senti-100k and ChnSentiCorp-htl-all.

	Positive Number	**Negative Number**
Weibo-Senti-100k	59,993	59,995
ChnSentiCorp-htl-all	5327	2439

In our experiment, the binary *BOW* vector was the model's input. Then, positive polarity clauses provided evidence on the presence of a class, while negative polarity clauses provided evidence on the absence of the class. Finally, TM sums scores from all clauses, and the class with the maximum score, were the target output. We used five-fold cross-validation to ensure the credibility of the experimental results. The dataset was randomly divided into 5 parts, four parts used for training and the other for testing). The scores were averaged as our final result.

We determined the optimal parameters of TM through extensive experiments, and set Clause number to 700, s to 3, Threshold to 9000. The symbol "-" represents those missing results which were not given in the original papers. For Weibo-senti-100k, we referred to previous work as baselines, and the evaluation results are shown in Table 2. For models not pre-trained, we used Sougou pre-trained word vectors to do word embedding, and set the batch size to 64. As shown in Table 2, TM outperformed all previous work in terms of both Accuracy and F1 score, and even outperformed some Bert-based ones.

Table 2. Experimental Result in Weibo-Senti-100k.

	Acc	**F1**
BiLSTM	88.98	88.89
TextCNN [10]	91.20	91.18
BERT [1]	96.74	97.31
ERINE [7]	96.75	97.30
CTBERT [30]	96.77	97.44
IAS-BERT [31]	98.10	98.10
BERT+MWA [32]	-	98.14
Tsetlin Machine	98.22	98.22

For evaluation in ChnSentiCorp-htl-all, we used Sougou pre-trained word vectors to do word embedding as input for models not pre-trained, and set the batch size to 64. For BERT, we used a base version pre-trained from Chinese which had 6-layer transformers, and set the batch size as 32.

As shown in Table 3, for F1-score, TM outperformed previous work, such as SINM and BAM, by one percentage, but was one percentage lower than the effect of BERT. We think this was because the extra knowledge learned by BERT through the pre-train mechanism was prominent in this scenario.

Table 3. Experimental Result in ChnSentiCorp-htl-all.

	Acc	**F1**
SVM	80.03	79.63
BiLSTM	83.42	83.20
TextCNN [10]	86.35	86.23
SINM [33]	-	89.40
BAM [34]	90.00	89.20
BERT [1]	92.80	91.29
Tsetlin Machine	90.20	90.17

4.2. Chinese Spam Review Detection

The newly proposed Douban movie reviews data set was used to evaluate the performance of our proposed model. The Douban movie review data set contains four main elements: review text, scores, review timing and the number of votes, meaning the number of people thinking a review is useful. As shown in Table 4, there were four kinds of features in this dataset, including review text features, review metadata features, user features and movie features. There were four classes in the data set: true and positive reviews, true and negative reviews, spam and positive reviews and spam and negative reviews. In general, the number of true reviews was about 5.7 times that of spam reviews, which gave an unbalanced distribution.

Table 4. Distribution of Douban Movie Review Dataset.

Class	Number of Reviews
True Positive	8511
True Negative	8349
Spam Positive	1337
Spam Negative	1537

It has been shown that fusion user and product level of features can improve detection performance [35]. Therefore, in this paper, we externally selected movie features and user features as additional features to help detect spam reviews. We did some data analysis and finally chose some features, as done by Cai et al. [36]. The features are listed in Tables 5 and 6. As is shown in Tables 5 and 6, we selected review length as an important feature, because it was found that the length of spam reviews was universally shorter than the length of true reviews, and the portion of short reviews in spam reviews was much larger than in true reviews. It is easy to understand that most spam writers do not deliberately fabricate longer reviews because it takes more time and effort, and they lack useful information. On the other hand, the frequency of review bursts was also an important feature that behaved differently between spam reviews and true reviews. For true reviews, the number pf reviews usually has a burst just after first release and gradually decreases as time goes by. However, spam reviews after the first release of a movie are possibly planned by the spammers to be little or flat. Similarly, the vote scores also exhibit some temporal dynamics. Just like movie burst, after the first burst, the scores possibly come from spammers and may exist in the low and middle levels. As done by Cai et al. [36], the standard deviation of score distribution was different between spam reviews and true reviews.

Table 5. Movie Features.

Movie Feature	Description
Max_s	Max vote score
Min_s	Min vote score
Avg_s	Average of vote scores
$High_s$	Portion of high vote scores
$Medium_s$	Portion of medium vote scores
Low_s	Portion of Low vote scores
$Max_{review\ per\ day}$	Max reviews per day
$Min_{review\ per\ day}$	Min reviews per day
Max_l	Max review length
Min_l	Min review length
Avg_l	Average review length
$Burst$	Number of review burst period
Std	Standard deviation of score distribution

In the experiment, the model's input was composed of three binary vectors of text, user and movie. Firstly, we used the BOW (bag-of-words) to convert the text into binary input. Secondly, we counted the number of different values of user and movie features and saved them in matrix U. Considering the actual situation, we set the maximum of the different number to 25. Finally, through the conversion mechanism mentioned above, the user and movie features were converted to the same binary form. The final input was the fusion of all these features. During learning, positive polarity clauses provided evidence on the presence of a class, while negative polarity clauses provided evidence on the absence of the class. Finally, TM summed the class score from all clauses, and the maximum of them was the final target.

Table 6. User Features.

User Feature	Description
Max_s	Max vote score
Min_s	Min vote score
Avg_s	Average of vote scores
$High_s$	Portion of high vote scores
$Medium_s$	Portion of medium vote scores
Low_s	Portion of Low vote scores
$Max_{review\ per\ day}$	Max reviews per day
$Min_{review\ per\ day}$	Min reviews per day
Max_l	Max review length
Min_l	Min review length
Avg_l	Average review length
P_{burst}	Proportion of reviews in review burst period

Models involved in the comparison included:

a. SVM: Support Vector Machine.

b. LR: Logistic Regression.

c. DPCN [37]: Deep Pyramid Convolutional Neural Networks(DPCNN).

d. Bi-LSTM: Bidirectional Long-Short Term Memory, learns contextual semantic information with multiple bidirectional LSTMs.

e. RCNN [38]: A combined model of RNN and CNN named Recurrent Convolutional Neural Networks (RCNN).

f. BERT+Attention: Cai et al. [36] used BERT to extract text features and designed a co-attention mechanism to combine movie and user features.

We determined the optimal parameters of TM through extensive experiments, and set Clause number to 700, s to 3, Threshold to 9000. The evaluation results are shown in Table 7. For models used in the comparsion, we used Sougou pre-trained word vectors as input, and set the batch size to 64.

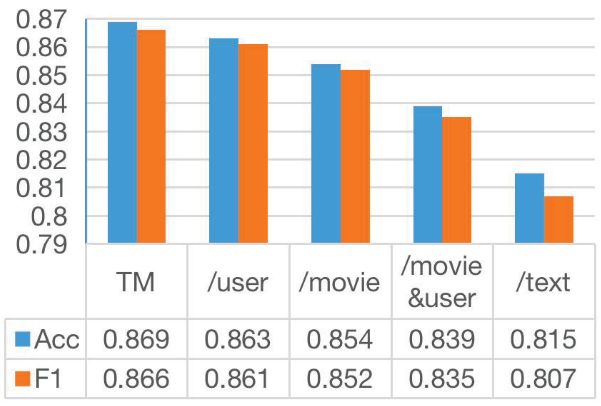
Table 7. Result of Spam Review Detection.

	Acc	**F1**
SVM	0.821	0.821
LR	0.823	0.814
DPCNN [37]	0.819	0.802
Bi-LSTM	0.823	0.818
RCNN [38]	0.815	0.804
BERT+Attention [36]	0.855	0.850
Tsetlin Machine	**0.869**	**0.866**

As can be seen from the Table 7, TM outperformed BERT with co-attention by more than 1.1 percent in Accuracy and in F1 score. It further outperformed all the other models by nearly 4 percent on accuracy and F1 score.

In order to show that the additional features we added were effective, we did some ablation experiments as well.

According to Figure 7, when the movie and user features were removed from input, the performance of the model decreased by about 3%, which showed that the introduction of additional features could effectively improve the model's ability to discriminate spam reviews. On the other hand, when text features were removed and only additional features utilized as input, the model's score was the lowest, with a drop of 6%. For the two additional features of users and movies, we found their differences according to the Figure 7. When only user features were removed at the input, the score dropped by about 1%. On the other hand, if only movie features were removed, the score dropped by 2%. We drew a conclusion that movie features had a greater impact than user features in our experiment.



	TM	/user	/movie	/movie &user	/text
🟦 Acc	0.869	0.863	0.854	0.839	0.815
🟧 F1	0.866	0.861	0.852	0.835	0.807

Figure 7. Ablation experiments of Spam Reviews Detection (/ means without).

4.3. Parameter Analysis

As shown in Table 8, the parameters that needed to be manually adjusted included: the number of clauses (C), the reward or punishment strength (s) during Feedback, and the threshold (T) for output summation.

Table 8. Description of Parameters.

Parameter	**Description**
C	The number of Clauses.
s	The strength of reward or punishment.
T	The threshold of sum value of scores.

In brief, C was used to extract semantic features, similar to the convolution kernels in CNN and the recurrent neurons in RNN. If C was too small, the judgment of the model was affected largely by individual clauses, leading to a large fluctuation and extreme situations. On the other hand, when C was too large, it caused resource redundancy and more calculation during the training process. A suitable parameter could bring better judgment and faster speed.

Hyperparameter s was similar to the learning rate and could affect the speed of convergence and the model's final effect as it had a big affect on the feedback process. When the model triggered Type I Feedback, s determined the probability of the model state transition. The larger the value of s, the more the TA were stimulated to include literals in their clauses. Clearly, the probability of encountering non-matching input grew as s increased (more literals were included). It meant that the Feedback was more likely to choose reward (penalty) operation. On the other hand, when s was small, the feedback mechanism gave Inaction feedback in a greater probability than reward and penalty. Therefore, the model was more inclined to maintain the existing state without changing.

T was highly related to C and it was the threshold for the summation of all the clauses in the output period. The introduction of T made the clauses distribute themselves according to the sub-patterns of data, so that T clauses of the correct polarity evaluated to 1 for any particular input X. Additionally, the summation target T opened up for interplay between the clauses, including rectification of special cases. The purpose of the mechanism was to ensure that the TM only spent a few of the available clauses to represent each specific sub-pattern. This was achieved by randomly selecting clauses to receive feedback, reducing intensity when approaching the summation target T.

Among them, C and T usually used empirical values [24], which were 700 and 9000, respectively, while s needed to be adjusted according to the task. In the experiment of sentiment analysis, we found that the choice of s had little effect on the convergence speed and final performance, and the model could easily reach the optimum in about two epochs. In the spam review detection task, we found that s had a greater impact on the convergence speed and the final effect. As shown in Figure 8, we drew the relationship between s and the convergence rate and final effect. There, the x-axis represents epoch number and y-axis represents weighted-F1 score.

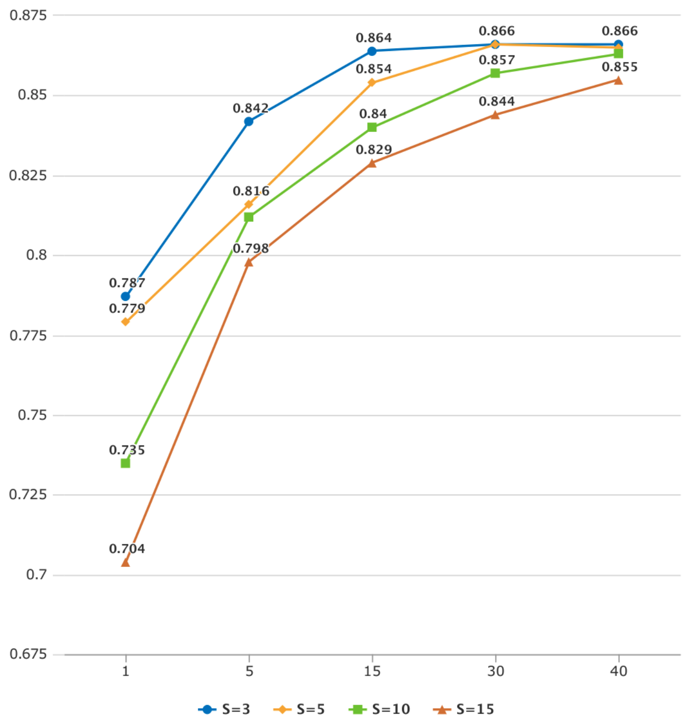


Figure 8. Change of Performance with Parameter s.

Through analysis, it can be seen that when $s = 3$, the model could converge quickly in no more than 15 epochs and achieved the best results. When s was configured as 15, the model needed more than 40 epochs to achieve similar performance.

4.4. Interpretability Analysis

We chose some examples to illustrate the learning process of TM on two tasks.

4.4.1. Chinese Sentiment Analysis

As Table 9 shows, we randomly selected some specific examples from datasets and translated them into English. Then, we illustrated the interpretability of TM model's learning processing by means of a case study.

Table 9. Samples and Translation.

Chinese text	0	肚子疼，不能出门去喝下午茶了[泪]
	1	好可爱的名字[可爱]
English translation	0	My stomach hurts, I can't go out to drink afternoon tea [tears].
	1	Such a cute name [cute].

Word segmentations are shown in Table 10. We depicted each class in different colors to highlight the differences.

Table 10. Segments of Samples.

Chinese text	0	肚子, 疼, 不能, 出门, 喝, 下午茶, 泪
	1	如此，可爱, 名字
English translation	0	stomach, hurt, can't, go out, drink, afternoon tea, tear
	1	such, cute, name

We tracked the positive and negative clauses during the training process, and printed out their judgment basis for different categories after training. The results are shown in Table 11. The first line in each clause was Chinese text and the second line in each clause was its English translation.

We can see that positive clauses chose the literals (red ones) of the target class (class 1) and negative clauses chose the literals (blue ones) from no-target classes (class 0).

Table 11. Literals learning results in Sentiment Analysis.

Clause+	如此 $\wedge$ 可爱 $\wedge$ 名字
	such $\wedge$ cute $\wedge$ name
Clause−	肚子 $\wedge$ 疼 $\wedge$ 不能 $\wedge$ 出门 $\wedge$ 喝 $\wedge$ 下午茶 $\wedge$ 泪
	stomach $\wedge$ hurt $\wedge$ can't $\wedge$ go out $\wedge$ drink $\wedge$ afternoon tea $\wedge$ tear

4.4.2. Chinese Spam Review Detection

Similar to steps above, we randomly select 4 Chinese texts (one for a class) from Douban Movie Review Dataset as shown in Table 12, and translate them into English.

Table 12. Samples from Douban Movie Review Dataset.

Chinese text	Spam Positive	全靠粉丝滤镜扛着
	True Positive	最棒的喜剧
	Spam Negative	恶心的片子，和编剧一样恶心
	True Negative	这尴尬的演技......
English translation	Spam Positive	It's all supported bt the fans' filter.
	True Positive	The best comedy.
	Spam Negative	Disgusting film, as disgusting as the screenwriter.
	True Negative	Awkward acting......

The results of word segmentation are in Table 13.

Table 13. Segments of Samples.

Chinese text	Spam Positive	全靠, 粉丝, 滤镜, 扛着
	True Positive	最棒, 喜剧
	Spam Negative	恶心, 片子, 编剧, 恶心
	True Negative	尴尬, 演技
English translation	Spam Positive	supported, by, fan, filter
	True Positive	best, comedy.
	Spam Negative	disgusting, movie, screenwriter
	True Negative	awkward, acting

We tracked 3 positive clauses ($Clause^+$) and 3 negative clauses ($Clause^-$) of class "Spam Positive" during training process, and printed out their judgment basis for different categories after training.

Among the 6 clauses, we indexed the three positive clauses $Clause^+$ as 1, 2, 3 and the three negative clauses $Clause^-$ as 1, 2, 3 as well. The results are shown in Table 14. We can see that positive clauses chose the literals of target class and negative clauses chose the literals from no-target classes.

Table 14. Literals learning results in Spam Review Detection.

Clause	**id**	**Chinese Text**	**English Translation**
Clause+	0	全靠 ∧ 粉丝 ∧ 滤镜 ∧ 扛着	supported ∧ by ∧ fan ∧ filter
	1	全靠 ∧ 粉丝 ∧ 滤镜 ∧ 扛着	supported ∧ by ∧ fan ∧ filter
	2	全靠 ∧ 粉丝 ∧ 滤镜 ∧ 扛着	supported ∧ by ∧ fan ∧ filter
Clause−	0	最棒 ∧ 喜剧	best ∧ comedy
	1	恶心 ∧ 片子 ∧ 喜剧	disgusting ∧ movie ∧ screenwriter
	2	尴尬 ∧ 演技	awkward ∧ acting

5. Conclusions

In this paper, we aspired to reduce the gap between the interpretability and performance for sentiment analysis and spam review detection in the Chinese language. We proposed a model binary input vector to classify the sentiment of a particular word in a sentence. We also converted some additional features from user and product into a binary form by a transformation mechanism and concatenated them with text input to enrich information. Such binary representations were then fed to a TM architecture that enjoyed a transparent learning process. Extensive experimental evaluations demonstrated the effectiveness of our model in Chinese over various baselines, and it even outperformed BERT etc. We demonstrated the interpretability and provided a clear picture of TM's learning process by means of a case study. This ends the description of our already done work, but our work is not finished. In the future, we will refine the results to address existing problems. For example, firstly, we will further enrich our work in the future, including proposing methods in more languages and trying to solve more NLP tasks, such as Named entity recognition (NER). Secondly, we will analyze the complexity, generalization and more effective feedback mechanism of the Tsetlin Machine in more detail.

Author Contributions: Conceptualization, X.Z., H.Z. and K.Y.; Methodology, X.Z.; Software, X.Z.; Validation, H.Z.; Formal analysis, X.Z.; Investigation, X.Z. and H.Z.; Resources, K.Y.; Writing—original draft, X.Z.; Writing—review & editing, H.Z., K.Y., X.W. and A.Y.; Visualization, X.Z.; Supervision, K.Y., X.W. and A.Y.; Funding acquisition, K.Y., X.W. and A.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This work was partially supported by the National Natural Science Foundation of China under Grant 61601046 and Grant 61171098, the 111 Project of China under Grant B08004, the BUPT Excellent Ph.D. Students Foundation under Grant CX2022149, and the EU FP7 IRSES Mobile Cloud Project under Grant 12212.

Data Availability Statement: Data is contained within the article.

Acknowledgments: This work was partially supported by the National Natural Science Foundation of China under Grant 61601046 and Grant 61171098, the 111 Project of China under Grant B08004, the BUPT Excellent Ph.D. Students Foundation under Grant CX2022149, and the EU FP7 IRSES Mobile Cloud Project under Grant 12212.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2018**, arXiv:1810.04805.
2. Granmo, O.C. The Tsetlin Machine–A Game Theoretic Bandit Driven Approach to Optimal Pattern Recognition with Propositional Logic. *arXiv* **2018**, arXiv:1804.01508.
3. Abeyrathna, K.D.; Bhattarai, B.; Goodwin, M.; Gorji, S.R.; Granmo, O.C.; Jiao, L.; Saha, R.; Yadav, R.K. Massively parallel and asynchronous Tsetlin Machine architecture supporting almost constant-time scaling. In Proceedings of the International Conference on Machine Learning, PMLR, Virtual, 18–24 July 2021; pp. 10–20.
4. Zhang, X.; Jiao, L.; Granmo, O.C.; Goodwin, M. On the Convergence of Tsetlin Machines for the IDENTITY-and NOT Operators. *IEEE Trans. Pattern Anal. Mach. Intell.* **2022**, *44*, 6345–6359. [CrossRef] [PubMed]
5. Jiao, L.; Zhang, X.; Granmo, O.C.; Abeyrathna, K.D. On the Convergence of Tsetlin Machines for the XOR Operator. *arXiv* **2021**, arXiv:2101.02547.
6. Yadav, R.K.; Jiao, L.; Granmo, O.C.; Goodwin, M. Interpretability in Word Sense Disambiguation using Tsetlin Machine. In Proceedings of the ICAART (2), Online, 4–6 February 2021; pp. 402–409.
7. Sun, Y.; Wang, S.; Li, Y.; Feng, S.; Chen, X.; Zhang, H.; Tian, X.; Zhu, D.; Tian, H.; Wu, H. Ernie: Enhanced representation through knowledge integration. *arXiv* **2019**, arXiv:1904.09223.
8. Zhang, X.; Zhou, H.; Yu, K.; Zhang, X.; Wu, X.; Yazidi, A. Sentiment Analysis for Chinese Dataset with Tsetlin Machine. In Proceedings of the 2022 International Symposium on the Tsetlin Machine (ISTM), Grimstad, Norway, 20–21 June 2022; pp. 1–6.
9. McCallum, A.; Nigam, K. A comparison of event models for naive bayes text classification. In Proceedings of the AAAI-98 Workshop on Learning for Text Categorization, Madison, WI, USA, 26–27 July 1998; Volume 752, pp. 41–48.
10. Dos Santos, C.; Gatti, M. Deep convolutional neural networks for sentiment analysis of short texts. In *COLING 2014, Proceedings of the 25th International Conference on Computational Linguistics: Technical Papers, Dublin, Ireland, 23–29 August 2014*; Dublin City University and Association for Computational Linguistics: Dublin, Ireland, 2014; pp. 69–78.
11. Xu, G.; Meng, Y.; Qiu, X.; Yu, Z.; Wu, X. Sentiment analysis of comment texts based on BiLSTM. *IEEE Access* **2019**, *7*, 51522–51532. [CrossRef]
12. Zulqarnain, M.; Ghazali, R.; Ghouse, M.G.; Mushtaq, M.F. Efficient processing of GRU based on word embedding for text classification. *JOIV Int. J. Inform. Vis.* **2019**, *3*, 377–383.
13. Jiao, Z.; Sun, S.; Sun, K. Chinese lexical analysis with deep bi-gru-crf network. *arXiv* **2018**, arXiv:1807.01882.
14. Zhou, C.; Sun, C.; Liu, Z.; Lau, F. A C-LSTM neural network for text classification. *arXiv* **2015**, arXiv:1511.08630.
15. Liu, G.; Guo, J. Bidirectional LSTM with attention mechanism and convolutional layer for text classification. *Neurocomputing* **2019**, *337*, 325–338.
16. Zhou, P.; Shi, W.; Tian, J.; Qi, Z.; Li, B.; Hao, H.; Xu, B. Attention-based bidirectional long short-term memory networks for relation classification. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), Berlin, Germany, 7–12 August 2016; pp. 207–212.

17. Yüksel, A.E.; Türkmen, Y.A.; Özgür, A.; Altınel, B. Turkish tweet classification with transformer encoder. In Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019), Varna, Bulgaria, 2–4 September 2019; pp. 1380–1387.
18. Fusilier, D.H.; Montes-y Gómez, M.; Rosso, P.; Cabrera, R.G. Detection of opinion spam with character n-grams. In *Computational Linguistics and Intelligent Text Processing, Proceedings of the CICLing 2015, Cairo, Egypt, 14–20 April 2015*; Part II 16; Springer: Cham, Switzerland, 2015; pp. 285–294.
19. Cagnina, L.C.; Rosso, P. Detecting deceptive opinions: Intra and cross-domain classification using an efficient representation. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.* **2017**, *25*, 151–174. [CrossRef]
20. Lau, R.Y.; Liao, S.; Kwok, R.C.W.; Xu, K.; Xia, Y.; Li, Y. Text mining and probabilistic language modeling for online review spam detection. *ACM Trans. Manag. Inf. Syst. (TMIS)* **2012**, *2*, 1–30. [CrossRef]
21. Ott, M.; Choi, Y.; Cardie, C.; Hancock, J.T. Finding deceptive opinion spam by any stretch of the imagination. *arXiv* **2011**, arXiv:1107.4557.
22. Yuan, C.; Zhou, W.; Ma, Q.; Lv, S.; Han, J.; Hu, S. Learning review representations from user and product level information for spam detection. In Proceedings of the 2019 IEEE International Conference on Data Mining (ICDM), Beijing, China, 8–11 November 2019; pp. 1444–1449.
23. Wu, Y.; Lian, D.; Xu, Y.; Wu, L.; Chen, E. Graph convolutional networks with markov random field reasoning for social spammer detection. *Proc. Aaai Conf. Artif. Intell.* **2020**, *34*, 1054–1061. [CrossRef]
24. Yadav, R.K.; Jiao, L.; Granmo, O.C.; Goodwin, M. Human-level interpretable learning for aspect-based sentiment analysis. *Proc. Aaai Conf. Artif. Intell.* **2021**, *35*, 14203–14212. [CrossRef]
25. Bhattarai, B.; Granmo, O.C.; Jiao, L. Measuring the Novelty of Natural Language Text Using the Conjunctive Clauses of a Tsetlin Machine Text Classifier. *arXiv* **2020**, arXiv:2011.08755.
26. Bhattarai, B.; Granmo, O.C.; Jiao, L. Explainable Tsetlin Machine framework for fake news detection with credibility score assessment. *arXiv* **2021**, arXiv:2105.09114.
27. Berge, G.T.; Granmo, O.C.; Tveit, T.O.; Goodwin, M.; Jiao, L.; Matheussen, B.V. Using the Tsetlin machine to learn human-interpretable rules for high-accuracy text categorization with medical applications. *IEEE Access* **2019**, *7*, 115134–115146.
28. Saha, R.; Granmo, O.C.; Goodwin, M. Mining Interpretable Rules for Sentiment and Semantic Relation Analysis using Tsetlin Machines. In Proceedings of the International Conference on Innovative Techniques and Applications of Artificial Intelligence, Cambridge, UK, 15–17 December 2020; pp. 67–78.
29. Yadav, R.K.; Jiao, L.; Granmo, O.C.; Goodwin, M. Enhancing Interpretable Clauses Semantically using Pretrained Word Representation. *arXiv* **2021**, arXiv:2104.06901.
30. Tang, F.; Nongpong, K. Chinese Sentiment Analysis Based on Lightweight Character-Level BERT. In Proceedings of the 2021 13th International Conference on Knowledge and Smart Technology (KST), Bangsaen, Chonburi, Thailand, 21–24 January 2021; pp. 27–32.
31. Zhang, L.; Lei, Y.; Wang, Z. IAS-BERT: An Information Gain Association Vector Semi-supervised BERT Model for Sentiment Analysis. In Proceedings of the International Conference on Cloud Computing, Qufu, China, 11–12 December 2020; pp. 31–42.
32. Li, Y.; Yu, B.; Xue, M.; Liu, T. Enhancing pre-trained Chinese character representation with word-aligned attention. *arXiv* **2019**, arXiv:1911.02821.
33. Li, G.; Zheng, Q.; Zhang, L.; Guo, S.; Niu, L. Sentiment Infomation based Model For Chinese text Sentiment Analysis. In Proceedings of the 2020 IEEE 3rd International Conference on Automation, Electronics and Electrical Engineering (AUTEEE), Shenyang, China, 20–22 November 2020; pp. 366–371.
34. Liu, W.; Cao, G.; Yin, J. Bi-level attention model for sentiment analysis of short texts. *IEEE Access* **2019**, *7*, 119813–119822.
35. Lim, E.P.; Nguyen, V.A.; Jindal, N.; Liu, B.; Lauw, H.W. Detecting product review spammers using rating behaviors. In Proceedings of the 19th ACM International Conference on Information and Knowledge Management, Toronto, ON, Canada, 26–30 October 2010; pp. 939–948.
36. Cai, H.; Yu, K.; Wu, X. Co-attention Based Deep Model with Domain-Adversarial Training for Spam Review Detection. In Proceedings of the 10th International Conference on Networks, Communication and Computing,Beijing, China, 10–12 December 2021; pp. 14–18.
37. Johnson, R.; Zhang, T. Deep pyramid convolutional neural networks for text categorization. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Vancouver, BC, Canada, 30 July–4 August 2017; pp. 562–570.
38. Lai, S.; Xu, L.; Liu, K.; Zhao, J. Recurrent convolutional neural networks for text classification. *Proc. Aaai Conf. Artif. Intell.* **2015**, *29*. [CrossRef]