*Proceedings of the 13th Conference on Language Resources and Evaluation (LREC 2022)*, pages 3761–3770
Marseille, 20-25 June 2022

# ConvTextTM: An Explainable Convolutional Tsetlin Machine Framework for Text Classification

**Bimal Bhattarai, Ole-Christoffer Granmo, Lei Jiao**
University of Agder
Norway
{bimal.bhattarai, ole.granmo, lei.jiao}@uia.no

**Abstract**

Recent advancements in natural language processing (NLP) have reshaped the industry, with powerful language models such as GPT-3 achieving superhuman performance on various tasks. However, the increasing complexity of such models turns them into "black boxes", creating uncertainty about their internal operation and decision-making. Tsetlin Machine (TM) employs human-interpretable conjunctive clauses in propositional logic to solve complex pattern recognition problems and has demonstrated competitive performance in various NLP tasks. In this paper, we propose ConvTextTM, a novel convolutional TM architecture for text classification. While legacy TM solutions treat the whole text as a corpus-specific set-of-words (SOW), ConvTextTM breaks down the text into a sequence of text fragments. The convolution over the text fragments opens up for local position-aware analysis. Further, ConvTextTM eliminates the dependency on a corpus-specific vocabulary. Instead, it employs a generic SOW formed by the tokenization scheme of the Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2019a). The convolution binds together the tokens, allowing ConvTextTM to address the out-of-vocabulary problem as well as spelling errors. We investigate the local explainability of our proposed method using clause-based features. Extensive experiments are conducted on seven datasets, to demonstrate that the accuracy of ConvTextTM is either superior or comparable to state-of-the-art baselines.

**Keywords:** Set-Of-Word (SOW), Convolutional Tsetlin Machine (CTM), Explainable, Human-Interpretable, Language Model, Text Classification, Tsetlin Machine (TM).

## 1. Introduction

Recent advances in Artificial Intelligence (AI) have brought increasingly accurate learning algorithms and powerful computation platforms. However, the accuracy gains come with escalating computation costs, and models are getting too complicated for humans to comprehend. Prohibitive computation costs hinder the training of state-of-the-art models, while a lack of model transparency reduces trust. When end-users cannot understand model decisions and researchers report on how susceptible the models are to data bias, confidence in machine learning models declines.

Mounting computation costs are problematic because of the long-term environmental impact. The obscurity of AI-driven decisions, however, raises immediate ethical concerns. Indeed, many decision errors can be fatal in high-risk application domains, including banking (Le and Viviani, 2018), medicine (Esteva et al., 2019), bioinformatics (Freitas et al., 2008), and self-driving cars (Badue et al., 2020). Transparency is required to ensure fair and safe decisions. AI model interpretability is thus a research area of increasing interest, crucial for further advancement of the AI field.

The field of Natural Language Processing (NLP) manifests all of the above concerns. With the development of powerful language models such as Transformers (Vaswani et al., 2017), the prediction accuracy already peaks for several NLP tasks, such as sentiment analysis and text classification. However, model complexity is immense, and only approximate interpretation techniques are available. On the other side of the spectrum, traditional models like Linear Regression, Logistic Regression, and Decision Trees can be highly interpretable but are far from achieving state-of-art accuracy. As such, they demonstrate the challenging accuracy-interpretability trade-off researchers must resolve.

The Tsetlin Machine (TM) (Granmo, 2018) is a rule-based pattern recognition approach that aims to bridge the gap between accuracy and interpretability. It uses a majority vote among multiple rules for decision-making. As such, TMs unify summation-based (cf. logistic regression) and rule-based (cf. decision trees) approaches. TMs further employ three powerful strategies for learning patterns: (i) Frequent pattern mining with so-called *Type I Feedback*; (ii) Pattern discrimination with so-called *Type II Feedback*; and (iii) Data dissection utilizing a *vote margin*.

Recent papers report TM accuracy results that are comparable to or surpass the accuracy of contemporary interpretable methods. These results cover sentiment analysis (Yadav et al., 2021a), novelty detection (Bhattarai et al., 2021a)(Bhattarai et al., 2022), fake news detection (Bhattarai et al., 2021b), and semantic relation analysis (Saha et al., 2020). Indeed, in some cases, the TM outperforms state-of-the-art deep learning baselines (Bhattarai et al., 2021b). Based on propositional logic and bitwise operations for learning, TMs offer an alternative approach to deep learning models. The advantages are interpretable learning, minimalist complexity, and comparable or better prediction accuracy in an increasing number of cases.

TMs achieve human-level interpretability through AND rules with negation that uses set-of-words (SOW) based features. In SOW, each word is one-hot encoded and then ORed into a document bit vector. One can also view each AND-rule as a conjunctive clause in propositional logic. A propositional variable represents each word. As reported by Blakely et al., clauses learned by a TM are highly discriminative, while being easy to interpret (Blakely and Granmo, 2020). Because each clause is self-contained, one can interpret the TM globally simply by inspecting each clause.

With increasing vocabulary size, however, the SOW representation becomes increasingly sparse. The sparseness affects both the preprocessing time, training time, and accuracy. Yadav et al. were able to reduce sparsity by boosting the SOW with GloVe synonyms (Yadav et al., 2021b). However, reducing pre-trained word embeddings to synonyms removes information from the embeddings. A main problem is that pre-trained word embeddings such as word2vec (Mikolov et al., 2013), GloVe (Pennington et al., 2014), and FastText (Bojanowski et al., 2017), are real-valued. TMs, however, require binary input. How to produce binary embeddings that are suitable for TMs is an open problem. Additionally, using non-interpretable word embeddings would impact TM interpretability.

**Contributions:** In this paper, we propose ConvTextTM, a novel convolutional TM (CTM) (Granmo et al., 2019) architecture for text classification. While legacy TM solutions treat the whole text as a corpus-specific SOW, ConvTextTM breaks down the text into a sequence of text fragments. The convolution over the text fragments opens up for local position-aware analysis. Further, ConvTextTM eliminates the dependency on a corpus-specific vocabulary. Instead, it employs a generic SOW formed by the tokenization scheme of the Bidirectional Encoder Representations from Transformers (BERT). The convolution binds together the tokens, allowing ConvTextTM to address the out-of-vocabulary problem as well as spelling errors. We investigate the local explainability of our proposed method using clause-based features. To our knowledge, this is the first time CTM has been used in a text-based application.

## 2. Related Work

Human interpretable machine learning models have a long history, with Breiman's research on decision trees and random forests being an early example (Breiman, 2001)(Li et al., 1984). Since then, several approaches for explaining the prediction of models have appeared. However, only a few studies focus on model interpretability (Lipton, 2018)(Lage et al., 2019). Some of these address local interpretability, i.e., they study how to understand single outputs (Ribeiro et al., 2016). Others try to interpret the entire model, providing global interpretability (Ustun and Rudin, 2016).

Gilpin et al. define interpretability of deep learning in terms of three categories (Gilpin et al., 2018): (1) models that map an input to output, focusing on connectivity, (2) models that explain latent variables for an explanation, and (3) models that explain themselves. Based on these categories, Agarwal et al. (Agarwal et al., 2020) proposed a deep learning-based interpretable model inspired by the family of Neural Additive Models (NAMs). However, their model was not able to surpass regular deep learning models accuracy-wise. As recent advances have demonstrated, interpretable rule-based approaches may provide a better trade-off between interpretability and accuracy. Numerous approaches, such as frequent itemset mining for association rule learning (Agrawal et al., 1993), and Probably Approximately Correct (PAC) (Feldman, 2009) have provided insights into how data patterns can be represented in Disjunctive Normal Form (DNF) using propositional formula. These techniques, however, are generally not scalable and robust to noise, and cannot compete with deep learning models.

TMs are a recent approach to rule-based machine learning that extracts human-interpretable patterns from data using propositional reasoning. Berge et al. (Berge et al., 2019) employed TM for categorizing medical records, with an emphasis on the interpretability of the rules generated by TM. They demonstrated that TMs are interpretable, yet provide competitive accuracy in comparison with other machine learning techniques, including Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) Neural Networks. Other researchers report similar findings, for example within regression (Darshana Abeyrathna et al., 2020), natural language understanding (Yadav et al., 2021a; Bhattarai et al., 2021b; Saha et al., 2020; Yadav et al., 2021c), and speech understanding (Lei et al., 2021). Additionally, the Convolutional TM performs competitively on MNIST, Fashion-MNIST, and Kuzushiji-MNIST, compared with several other contemporary methods including CNNs (Granmo et al., 2019).

Methods for global and local model interpretation are gaining increasing attention in the NLP research community. (Jacovi et al., 2018) examined how CNN networks process and categorize text. They demonstrate, by employing various alternative activation patterns, how filters may capture several semantic classes of n-grams. They compute word-level scores in two ways: locally and globally. Both techniques, however, need a specific input and employ leave-one-out evaluations. The authors of (Xiong et al., 2018) conclude that Layer-wise Relevance Propagation (LRP) extracts more relevant features and argue that a single word does not adequately explain any given outcome. For instance, a single word can be a component of a significant n-gram. However, their method requires knowledge of the model parameters, which cannot be consid-

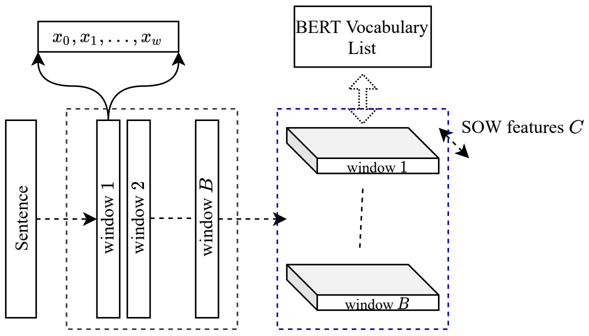


Figure 1: Preprocessing text for Convolutional TM.

ered an explanation for the black-box nature of the neural network. (Chen and Ji, 2020) use masks over word embedding named variational word mask (VMASK) and provide an interpretable architecture for text classification to learn task-specific important words. They demonstrate the model's interpretability by comparing it to LIME and SampleShapley (Strumbelj and Kononenko, 2010). Similarly, (Nguyen, 2018) presented a robust technique for text categorization that relies on local explanation. They demonstrate through crowdsourcing experiments that automatic evaluation metrics based on word elimination, such as LIME, are reasonably consistent with human evaluations.

Despite these advances in NLP interpretability, strong transformer-based language models such as BERT (Devlin et al., 2019a) and XLNet (Yang et al., 2019a) outperform alternative approaches that rely on pre-trained embeddings. In summary, previous work report that as the complexity of the model increases, the interpretation becomes more ambiguous (Lipton, 2018). For example, it is possible that the small neural network models may be more interpretable than large decision trees. One can argue that some models employ an attention mechanism that can be interpreted based on the weighting of the input features. Still, other researchers report that attention weights do not provide a meaningful explanation (Bai et al., 2020)(Serrano and Smith, 2019). To overcome these shortcomings, the interest in rule-based interpretable techniques is rising, with TM being a promising approach. However, as discussed, the SOW sparsity introduced by large vocabularies is significantly limiting TMs. Yadav et al. were able to partially mitigate increasing sparsity by boosting TM SOWs with GloVe synonyms, significantly increasing accuracy (Yadav et al., 2021b). The work we present here, however, attempts to address this issue by employing a fixed-sized pre-trained BERT embedding vocabulary. Accordingly, we establish a general TM SOW of manageable size.

## 3. ConvTextTM Architecture

### 3.1. Tsetlin Machine Basics

A basic TM consists of two-action Tsetlin Automata (TAs) with $2N$ states. Each TA performs an action associated with its current state, which is either an "Include" action (in State 1 to $N$) or an "Exclude" action ($N+1$ to $2N$). The state updates are based on iterative feedback, i.e., rewards and penalties. Rewards reinforce the action performed by the TA, while penalties suppress the action. In this way, the TA progressively shifts toward the optimal action (Tsetlin, 1961). To address the pattern recognition problem, the collection of TAs construct conjunctive clauses in propositional logic. The input to a TM consists of propositional values, i.e., the input is Boolean. A TM uses both the non-negated and negated version of each input when forming the clauses, each of which is handled by an individual TA. Accordingly, each TM clause captures a specific sub-pattern. The output, in turn, is determined based on votes for the available classes. It is the class with the largest number of matching sub-patterns that wins the round of voting and is output by the TM.

The TM takes a Boolean input vector, $X = (x_1, \ldots, x_o)$, $x_k \in \{0,1\}$, $k \in \{1, \ldots, o\}$ to be classified into one of the $Cl$ classes, $Y = (y_1, \ldots, y_{Cl})$, where $Cl$ is total number of classes. From the input vector, we obtain $2o$ literals $L = (l_1, l_2, \ldots, l_{2o})$. The literals consist of the inputs $x_k$ and their negated counterparts $\bar{x}_k = \neg x_k = 1 - x_k$, i.e., $L = (x_1, \ldots, x_o, \neg x_1, \ldots, \neg x_o)$. The TM patterns are formed using $m$ conjunctive clauses. Subscript $j = 1, \ldots, m/2$ denotes the clause index, while the superscript describes the *polarity* of a clause. In brief, half of the clauses are assigned positive polarity, i.e., $C_j^+$, and the other half are assigned negative polarity, i.e., $C_j^-$. A clause $C_j^\xi$, $\xi \in \{-,+\}$, is formed by ANDing a subset of the literals, i.e., the ones that are "included" by the corresponding TAs, $L_j^\xi \subseteq L$, as:

$$C_j^\xi(X) = \bigwedge_{l_k \in L_j^\xi} l_k = \prod_{l_k \in L_j^\xi} l_k. \tag{1}$$

The final classification output is obtained by subtracting the negative votes from the positive votes, and then thresholding the resulting sum via the unit step function $u$:

$$\hat{y} = u\left(\sum_{j=1}^{m/2} C_j^+(X) - \sum_{j=1}^{m/2} C_j^-(X)\right). \tag{2}$$

The TM learning process carefully guides the TAs to make optimal decisions. The reinforcement is channeled directly to the conjunctive clauses. Each clause, in turn, forwards the feedback to its individual TAs randomly. To this end, the TM employs Type I and Type II feedback. These feedback types control how rewards and penalties are distributed to the TAs. The distribution depends on six factors: (1) target output ($y = 0$ or $y = 1$), (2) clause polarity, (3) clause output ($C_j = 0$ or 1), (4) literals value ($x = 1$, or $\neg x = 1$), (5) vote sum, and (6) the current state of the TA. Type I feedback is intended to generate frequent patterns, eliminate false

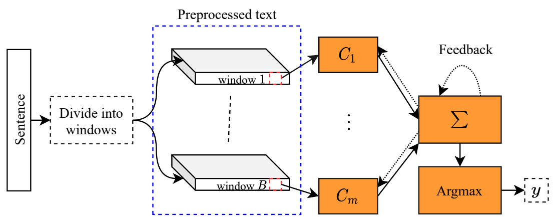


Figure 2: ConvTextTM Architecture.

| Dataset name | Train set size | Test set size | Label |
|---|---|---|---|
| PolitiFact | 716 | 238 | 2 |
| GossipCop | 15,175 | 5,058 | 2 |
| BBCSports | 517 | 220 | 5 |
| Twitter | 2,176 | 932 | 3 |
| Query | 17,500 | 3,850 | 2 |
| R8 | 5,478 | 2,189 | 8 |
| WOS-5736 | 4,588 | 1,148 | 11 |

Table 1: Dataset Statistics.

negatives, and make clauses evaluate to 1. Positive polarity clauses receive Type I input when $y = 1$, while negative polarity clauses receive it when $y = 0$. Type II feedback, on the other hand, improves pattern discrimination, suppresses false positives, and spurs clauses to evaluate to 0. Type II feedback is given to positive polarity clauses when $y = 0$ and to negative polarity clauses when $y = 1$.
The feedback is further regulated by the sum of votes $v$ for each output class. That is, the voting sum is compared against a voting margin $T$, which is employed to guide distinct clauses to learn different sub-patterns. The details of the learning process can be found in (Granmo, 2018).

### 3.2. Convolutional Tsetlin Machine and Pre-processing for Text

The CTM is a variation of TM developed for image classification. The CTM takes an image $X = (x_k) \in \{0,1\}^{\mathcal{A}\times\mathcal{B}\times\mathcal{C}}$ as input. Here, $\mathcal{A}\times\mathcal{B}$ specifies the image dimension and $\mathcal{C}$ denotes the depth of the binary layers. Because CTM is designed for fixed-sized images, we must preprocess our text data prior to feeding it to the CTM. To begin with, each input instance is segmented into a number of equal-sized windows. To make the window size equal, we append "PAD" tokens at the end of the instances. Because the average lengths of instances in different datasets can be quite different, the number of windows generated for different datasets may vary.
To integrate the information from BERT, we create a dictionary that indexes all of the tokens in the BERT vocabulary. The words in each window are then one-hot encoded using their indices in the dictionary and the window number in which they appear. That is, each word is denoted by a $2D$ SOW representation. For example, suppose a word present in the second window of the first input instance is assigned the $500^{th}$ index in the dictionary of BERT. Then, position *(2, 500)* of the input is set to 1. After preprocessing, each input text is represented as a Boolean $2D$ SOW. Now, the complete input dataset can be represented by $(x_k) \in \{0,1\}^{\mathcal{A}\times\mathcal{B}\times\mathcal{C}}$. Here $\mathcal{A}$ denotes the total number of input instances, $\mathcal{B}$ denotes the total number of windows, and $\mathcal{C}$ represents the total number of tokens in the BERT vocabulary list. Figure 1 depicts the full preprocessing pipeline.
The CTM for text uses clauses with spatial dimension $1 \times 1$ as filters to perform convolution over the preprocessed input text. The filter evaluates $P = \frac{\mathcal{B}-1}{q} + 1$ patches of dimensionality $C$, with $q$ being the step size of the convolution. Additionally, each clause can specify binary encoded coordinates for location awareness (visualized as red boxes in Figure 2). The coordinates are encoded by augmenting each patch input vector $\mathbf{x}^p = [x_k^p] \in \{0,1\}^{\mathcal{C}}$, $p \in \{1, 2, \ldots, P\}$, with a coordinate vector. The coordinate vector $\mathbf{l}^p = [l_k^p] \in \{0,1\}^{\mathcal{P}\times 2}$ of each patch is threshold-encoded (Darshana Abeyrathna et al., 2020). That is, the input vector is extended with one propositional variable per position. The resulting augmented vector is denoted by $\mathbf{x}^p : x^p = [x_k^p] \in \{0,1\}^{\mathcal{C}+P}$. The overall ConvTextTM architecture is depicted in Figure 2.
Classification in CTM is similar to that of vanilla TM, as shown in Eq. 2. However, because we have $P$ input vectors $x^p$ for CTM, each clause produces $P$ output values. The final output is obtained by OR*ing* the individual outputs from each patch, specified as follows:

$$\hat{y} = u\left[\sum_{j=1}^{m/2} \bigvee_{p=1}^{P} \left(\bigwedge_{l_k \in L_j^+} l_k^p\right) - \sum_{j=1}^{m/2} \bigvee_{p=1}^{P} \left(\bigwedge_{l_k \in L_j^-} l_k^p\right)\right]. \tag{3}$$

CTM learning leverages the TM learning, with clauses receiving the feedback based on the input vector and label. However, for CTM, there are $P$ input patches in an input. Therefore, when a clause is updated, CTM randomly selects one of the patch input vector $x^p$ that made the clause evaluate to 1:

$$\mathbf{x}_j^p = Random\left(\left\{\mathbf{x}^p \middle| \left(\bigwedge_{l_k \in L_j^{\xi}} l_k^p\right) = 1, 1 \le p \le P\right\}\right). \tag{4}$$

The randomly chosen patch input $x_p$ enables each clause to extract a specific sub-pattern, and the unpredictability of the uniform distribution statistically scatters the clauses for various sub-patterns in the target text.

## 4. Experiments

### 4.1. Datasets

Table 1 provides an overview of the publicly available datasets used in our experiment. The goal of this study is to assess our framework across a variety of datasets with varying semantic content, sentence lengths, and

| Datasets | Epochs | #clauses | Threshold (T) | Sensitivity (s) | Window |
|---|---|---|---|---|---|
| PolitiFact | 200 | 10,000 | 100*100 | 10.0 | 3 |
| GossipCop | 200 | 15,000 | 100*100 | 20.0 | 3 |
| BBCSports | 100 | 10,000 | 150*150 | 10.0 | 2 |
| Twitter | 100 | 10,000 | 50*50 | 10.0 | 2 |
| Query | 100 | 5,000 | 150*150 | 10.0 | 2 |
| R8 | 150 | 10,000 | 100*100 | 10.0 | 2 |
| WOS-5736 | 150 | 10,000 | 150*150 | 10.0 | 2 |

Table 2: Hyperparameter configurations.

linguistic patterns. To this end, we employ datasets related to fact-checking, document classification, academic text classification, and short text classification. We organize the datasets as follows:

- **Fact Checking**: We here utilize a publicly available data repository for fake news detection, FakeNewsNet (Shu et al., 2020a). The labels are provided by professional journalists from the fact-checking websites *PolitiFact* and *GossipCop*. PolitiFact focuses on U.S. political news, whereas GossipCop focuses on entertainment news from various media.

- **Topic Classification**: For topic classification, we use BBC Sports and a Twitter dataset. BBC Sports consists of 737 documents from the BBC Sports website, collected between 2004 and 2005, labeled according to five sports categories. Furthermore, the Twitter dataset consists of a collection of $5,513$ tweets that have been hand-classified into one of the four topics: Apple, Google, Twitter, and Microsoft. The tweets are further labeled with sentiments positive, negative, or neutral.

- **Short text classification**: These datasets involve short text sentences. We explore performance using the Query (Faruqui and Das, 2018) and R8 (Debole and Sebastiani, 2005) datasets. Google's query dataset was created by crowdsourcing well-formedness annotations for $25,100$ queries from the Paralex corpus. Each query was binary annotated by five raters regarding whether the query was well-formed. R8 is a subset of the Reuters-21578 corpus that contains news documents classified into the eight most popular classes. Note that the document distribution in R8 is severely skewed, with the smallest class holding just 51 documents.

- **Academic text classification**: We utilize the Web of Science (WoS)-5736 datset (Kowsari et al., 2017a), which comprises $5,736$ published papers with eleven categories organized under three main categories.

## 4.2. Baselines

We begin by summarizing the baseline methods.

- FastText (Joulin et al., 2017) uses a linear classifier to input a document embedding, which is created by the average of word/n-grams embeddings.

- Rhetorical Structure Theory (RST) (Rubin et al., 2015) uses a tree structure to depict the relationship between words in a document. It extracts the news style features from a bag of words by mapping them into a latent feature representation.

- Linguistic Inquiry and Word Count (LIWC) (Pennebaker et al., 2015) is used to extract and learn features from psycholinguistic and deception categories.

- Hierarchical attention neural network (HAN) (Yang et al., 2016) is employed for embedding word-level attention on each sentence and sentence-level attention on news content for fake news detection.

- CNN-text (Kim, 2014) utilizes a convolutional neural network (CNN) with pre-trained word vectors to perform sentence-level classification. Multiple convolutional filters enable the model to collect various granularities of text features from news articles.

- LSTM-ATT (Lin et al., 2019a) employs long short term memory (LSTM) in conjunction with an attention mechanism. The model feeds a two-layer LSTM with a 300-dimensional vector representation of news items as input for fake news detection.

- Hierarchical Deep Learning for Text Classification (HDLTex) (Kowsari et al., 2017b) performs hierarchical classification by stacking deep learning architectures to give specialized comprehension at each level of the document hierarchy.

- RoBERTa-MWSS was proposed in (Shu et al., 2020b), which employs the Multiple Sources of Weak Social Supervision (MWSS) approach built upon RoBERTa (Liu et al., 2019).

- Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2019b) is a Transformer-based model which contains an encoder with 12 transformer blocks, self-attention heads, and a hidden shape size of 768.

- XLNet (Yang et al., 2019b) is a generalized autoregressive pretraining model that integrates autoencoding and a segment recurrence mechanism from transformers.

Other approaches include Word Mover's Distance (WMD), which defines the distance between two documents as the minimum cost of converting the words of one document into words of the other. Deepsets adopt objective functions defined on sets that are invariant to permutations, allowing for the building of a deep network architecture that operates on sets. Similarly, NNattention and Set-Transformer are variants of Deepsets that substitutes an attention mechanism for the sum operator.

| Datasets | RST | LIWC | HAN | CNN-text | LSTM-ATT | RoBERTa-MWSS | BERT | XLNet | $TM$ | $TM_{conv}$ | $TM_{conv}(max)$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| PolitiFact | 60.7 | 76.9 | 83.7 | 65.3 | 83.3 | 82.5 | 88 | 89.5 | 87.1±0.24 | 90.27±0.33 | **91.21** |
| GossipCop | 53.1 | 73.6 | 74.2 | 73.9 | 79.3 | 80.3 | 85 | 85.5 | 84.2±0.03 | 85.82±0.27 | **86.28** |

Table 3: Performance comparison of our model with other baseline models for fact checking.

| Datasets | WMD | Deepsets | NNattention | Transformer | LSTM | BERT | XLNet | $TM$ | $TM_{conv}$ | $TM_{conv}(max)$ |
|---|---|---|---|---|---|---|---|---|---|---|
| BBCSports | 95.40± 0.70 | 74.55±20.1 | 95.28±0.97 | 95.82±1.23 | 95.52 | **99** | 98 | 96.91 | 96.78 ± 0.32 | 97.97 |
| Twitter | 71.3±0.70 | 70±1.62 | 70.91±0.62 | 72.21±0.47 | 72.1 | 74.71 | **78** | 71.13 | 70.67±0.27 | 71.91 |

Table 4: Performance comparison of our model with other baseline models for topic classification.

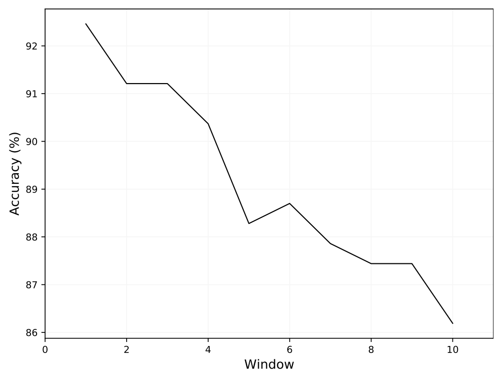


Figure 3: Accuracy *vs* Window.

### 4.3. Training and Testing

We use a random train-test split of 70% / 30% for all datasets except R8 and query as they have predefined train and test sets. The process is repeated five times, and the average accuracy and F1 scores are reported. To provide robust results, we calculated an ensemble average by first taking the average of 50 stable epochs, followed by taking the average of the resulting five averages. The experiments were conducted on the server - NVIDIA DGX-2 with dual Intel Xeon Platinum 8168, 2.7 GHz, 16× NVIDIA Tesla V100 (32 GB), and Ubuntu 18.04 LTS x64. The hyperparameter configurations of the experiment for each dataset are listed in Table 2.

## 5. Results and Discussions

In this section, we evaluate the performance of our framework with the aforementioned baselines for each classification task, including seven datasets.

For fact-checking (i.e., PolitiFact and GossipCop), we see that HAN outperforms LIWC, CNN-text, and RST on both datasets. This is arguable because HAN can capture the syntactic and semantic rules required to detect fake news via hierarchical attention. Similarly, the LIWC performs better than RST. One possible reason for this is that LIWC is able to extract the linguistic features from news articles based on words that denote psycholinguistic characteristics. The LSTM-ATT, which incorporates substantial preprocessing utilizing count features and sentiment features, as well as hyperparameter tweaking (Lin et al., 2019b), performs similarly to HAN in PolitiFact but exceeds it on GossipCop. One possible explanation for this can be that the attention mechanism is able to capture the relevant representation of the input. Our model outperforms all the baselines, including transformer-based models such as RoBERTa-MWSS, BERT, and XLNet. Indeed, ConvTextTM achieves maximum accuracy of 91.21% for PolitiFact and 86.28% for GossipCop. Besides, our model is significantly simpler than the deep learning models since we do not use any pre-trained embeddings in preprocessing. This also contributes to the transparency, interpretability, and explainability of our model. Table 3 illustrates the comparison outcomes.

For topic classification (i.e., BBC Sports and Twitter), Table 4 shows the classification accuracy of our proposed model and those of the baselines. The set-transformer outperforms other deep learning algorithms including Deepsets, NNattention, and LSTM, marginally. The disparate performance of LSTM might be explained by the fact that bidirectional LSTMs are ineffective in modeling ungrammatical text. However, BERT and XLNet surpass all other approaches, with BERT achieving the accuracy of 99% in BBC Sports. As the prior knowledge in the pre-trained BERT and XLNet is not particular to any domain, fine-tuning is required to unleash the actual potential of BERT and XLNet. Our proposed CTM method outperforms the vanilla TM and all other techniques except BERT and XLNet, and obtains an accuracy close to those of BERT and XLNet. The low accuracy of our model in the Twitter dataset is most probably because our model, by relying entirely on the SOW approach, overlooks important word orders that are useful in sentiment classification.

For short text classification (i.e., Query and R8), Table 5 compares the accuracy of ConvTextTM against state-of-the-art methods. As previously stated, with large-scale pretraining and fine-tuning on small-data tasks, BERT achieves 80% on Query and XLNet obtains 98% on R8, surpassing other approaches significantly. Human evaluation was conducted on the query dataset by submitting $1,000$ queries to a human proficient in English, who were able to match the label with 88.4% accuracy, indicating an estimated upper bound.

For academic text classification (i.e., WOS-5736), Table 6 compares the accuracy of our model with other techniques. While the RNN outperforms DNN and CNN, HDLTex with stacked deep learning CNN architecture performed surprisingly well, outperforming

| Datasets | CNN | Fasttext | LSTM | HAN | BERT | XLNet | Human | $TM$ | $TM_{conv}$ | $TM_{conv}(max)$ |
|---|---|---|---|---|---|---|---|---|---|---|
| Query | 67.38 | 62.1 | 65.8 | 64.64 | 80 | 77 | **88.4** | 53.87 | 67.43±0.22 | 67.94 |
| R8 | 95.71 | 96.13 | 96.09 | - | 97.8 | **98** | - | 96.16 | 96.32±0.11 | 96.43 |

Table 5: Performance comparison of our model with other baseline models for short text classification.

| Dataset | DNN | CNN | RNN | Stacking SVM | HDLTex-CNN | BERT | XLNet | $TM$ | $TM_{conv}$ | $TM_{conv}(max)$ |
|---|---|---|---|---|---|---|---|---|---|---|
| WOS-5736 | 86.15 | 88.68 | 89.46 | 85.68 | 90.93 | 90.24 | 90 | 89.47 | 90.73±0.27 | **91.28** |

Table 6: Performance comparison of our model with other baseline models for academic classification.

BERT, XLNet, and TM. However, the maximum accuracy from our proposed model outperformed HDLTex-CNN by a slight margin.

### 5.1. Accuracy vs Window Size

We now use the PolitiFact dataset to investigate the effect of increasing the window size $\mathcal{W}$ on accuracy. As PolitiFact contains long sentences per example, our model is able to achieve very high accuracy by utilizing clauses to capture linguistic patterns in each window. However, for datasets with short sentences such as Query and R8, it is not feasible to use more than two windows since each window would contain only a few words, rendering it useless for classification. Figure 3 illustrates the trade-off between window size and accuracy. We notice that as the window size increases, the accuracy decreases. The highest accuracy is attained when $\mathcal{W} = 1$, which means that the input text sentence has not been divided into any windows. The maximum accuracy obtained for each dataset using $\mathcal{W} = 1$ is shown in Table 8. Because we are investigating the local interpretability using parts of a text sentence, we have to break the text sentence into windows. As a result, $\mathcal{W} = 2$ and $\mathcal{W} = 3$ are most suitable for us, as their accuracy is comparable to that of $\mathcal{W} = 1$, with the added benefit of local interpretation. From this experiment, we can conclude that the length of the text sentence determines the window size. That is, if the dataset contains lengthy sentences, we may increase the window size without compromising accuracy.

## 6. Interpretability Analysis

We examine the interpretability of our framework using the R8 dataset. Here, we consider both global and local interpretability. We define global interpretability as identifying global features from the trained model that contribute significantly to classification. In contrast, local interpretability refers to identifying significant features inside a particular window. For example, consider the following input sentence from the R8 dataset, *"Taiwan shipbuilder looks for Japanese ventures Taiwan's state-owned China shipbuilding corp csbc plans to seek joint production agreements with Japan and further diversify into ship repairing."* with the classification label "ship". We divide the sentence into windows. Using a window size of two, the sentence can be separated into two phrases: **Window 1**: *[Taiwan shipbuilder looks for Japanese ventures Taiwan's state-owned China shipbuilding corp]* and **Window 2**: *[csbc plans to seek joint production agreements with Japan and further diversify into ship repairing]*. As a result, we can observe that the first window is sufficient to classify the text. Therefore, through local interpretability, our proposed model determines which section of the sentence is relevant for classification. Additionally, global interpretability can be used to highlight the relevant features inside the window as seen in Table 7.

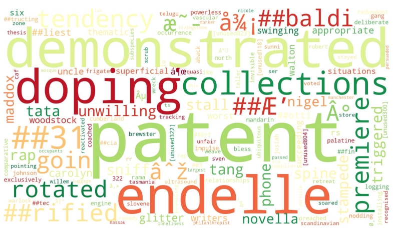
We split the R8 dataset into two windows for our experiment, given the dataset contains short sentences. That is, as seen in Figure 1, each example sentence is split into two local parts. The model is trained on the preprocessed dataset, and clause-specific features are extracted. We note that certain clauses capture features of all the windows, whereas others capture features of a particular window. This suggests that we can distinguish between features that have a global impact and the ones that are responsible for a particular classification on a local location-aware level. To demonstrate the local interpretability, we visualize the features from two windows for "**acq**" class in Figure 4. The features in each window signify the classification decision based on the corresponding part of the sentence. That is, we can know which part of the sentence is essential for the classification.

## 7. Conclusion

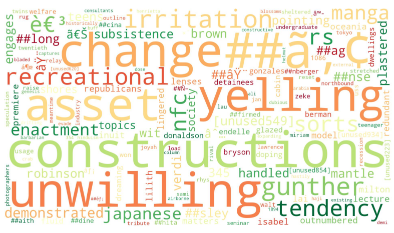
In this paper, we introduce an explainable Convolution Tsetlin Machine (CTM) architecture for text classification. Due to the fact that CTM is optimized for image-based tasks, we preprocess the text document into various windows. Thus, the clauses from the CTM can capture the semantic features associated with each particular window's word patterns. We propose using predefined fixed-length vocabulary features derived from the standard Bidirectional Encoder Representations from Transformers (BERT) tokens. This allows us to mitigate the TM dependency on vocabulary for each dataset and the out-of-vocabulary problem. We illustrate the effectiveness of our model on four classification tasks using seven different datasets. Our experimental findings demonstrate that our approach is superior to or competitive with more sophisticated and non-transparent approaches, including BERT and XLNet. We then investigate the effect of different window

| acq | crude | earn | grain | interest | money-fx | ship | trade |
|---|---|---|---|---|---|---|---|
| pointing | ##connect | accomplish | island | tesla | kruger | farmhouse | ##ities |
| doping | thorns | ##sby | treating | ##oko | ashton | road | harlow |
| ##rified | notably | looting | surface | linux | tonight | agency | pianos |
| evergreen | ##enter | ##osed | facebook | bed | ##sitor | ##voking | sanskrit |
| phone | dramatic | endelle | trail | slams | becker | ##met | plunged |
| premiere | chloride | confrontation | ##emia | photo | bed | rockies | ##cured |
| demonstrated | racers | temporarily | summer | mural | burning | trafficking | script |
| collections | fountain | werewolf | road | vaccines | handicap | likeness | theories |
| unwilling | ##lon | segregation | garbage | families | lightning | flaming | ineffective |
| ##liest | safety | presiding | ##nity | bronze | patent | gesellschaft | extensive |

Table 7: Significant features captured by our model for each class in R8 dataset.



(a) Window 0.



(b) Window 1.

Figure 4: Wordcloud visualization of local interpretability for "**acq**".

| Dataset | Accuracy (max) |
|---|---|
| PolitiFact | 92.46 |
| GossipCop | 85.94 |
| BBCSports | 99.54 |
| Twitter | 73.67 |
| Query | 68.25 |
| R8 | 96.93 |
| WOS-5736 | 92.42 |

Table 8: Maximum accuracy obtained using window size 1 (i.e., $\mathcal{W} = 1$).

sizes on our model's performance. Additionally, we illustrate the global and local interpretability provided by the CTM depending on the significance of features captured by clauses in separate windows.

## 8. Bibliographical References

Agarwal, R., Frosst, N., Zhang, X., Caruana, R., and Hinton, G. E. (2020). Neural additive models: Interpretable machine learning with neural nets. *arXiv preprint:2004.13912*.

Agrawal, R., Imieliński, T., and Swami, A. (1993). Mining association rules between sets of items in large databases. In *Proceedings of ACM SIGMOD international conference on Management of data*, pages 207–216.

Badue, C., Guidolini, R., Carneiro, R. V., Azevedo, P., Cardoso, V. B., Forechi, A., Jesus, L., Berriel, R., Paixao, T. M., Mutz, F., et al. (2020). Self-driving cars: A survey. *Expert Systems with Applications*, page 113816.

Bai, B., Liang, J., Zhang, G., Li, H., Bai, K., and Wang, F. (2020). Why is attention not so attentive? *arXiv preprint:2006.05656*.

Berge, G. T., Granmo, O.-C., Tveit, T. O., Goodwin, M., Jiao, L., and Matheussen, B. V. (2019). Using the tsetlin machine to learn human-interpretable rules for high-accuracy text categorization with medical applications. *IEEE Access*, 7:115134–115146.

Bhattarai, B., Granmo, O.-C., and Jiao, L. (2021a). Measuring the novelty of natural language text using the conjunctive clauses of a Tsetlin machine text classifier. In *Proceedings of ICAART*.

Bhattarai, B., Granmo, O.-C., and Jiao, L. (2021b). Explainable tsetlin machine framework for fake news detection with credibility score assessment. *arXiv preprint:2105.09114*.

Bhattarai, B., Granmo, O.-C., and Jiao, L. (2022). Word-level human interpretable scoring mechanism for novel text detection using tsetlin machines. *Applied Intelligence*.

Blakely, C. D. and Granmo, O.-C. (2020). Closed-form expressions for global and local interpretation of tsetlin machines with applications to explaining high-dimensional data. *arXiv preprint:2007.13885*.

Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the ACL*, 5:135–146.

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

Chen, H. and Ji, Y. (2020). Learning variational word masks to improve the interpretability of neural text classifiers. In *Proceedings of EMNLP*.

Darshana Abeyrathna, K., Granmo, O.-C., Zhang, X., Jiao, L., and Goodwin, M. (2020). The regression tsetlin machine: a novel approach to interpretable nonlinear regression. *Philosophical Transactions of the Royal Society A*, 378(2164):20190165.

Debole, F. and Sebastiani, F. (2005). An analysis of the relative hardness of reuters-21578 subsets. *Journal of the American Society for Information Science and technology*, 56(6):584–596.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019a). Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019b). Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*.

Esteva, A., Robicquet, A., Ramsundar, B., Kuleshov, V., DePristo, M., Chou, K., Cui, C., Corrado, G., Thrun, S., and Dean, J. (2019). A guide to deep learning in healthcare. *Nature medicine*, 25(1):24–29.

Faruqui, M. and Das, D. (2018). Identifying well-formed natural language questions. In *Proceedings of EMNLP*, pages 798–803.

Feldman, V. (2009). Hardness of approximate two-level logic minimization and pac learning with membership queries. *Journal of Computer and System Sciences*, 75(1):13–26.

Freitas, A. A., Wieser, D. C., and Apweiler, R. (2008). On the importance of comprehensible classification models for protein function prediction. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 7(1):172–182.

Gilpin, L. H., Bau, D., Yuan, B. Z., Bajwa, A., Specter, M., and Kagal, L. (2018). Explaining explanations: An overview of interpretability of machine learning. In *Proceedings of IEEE DSAA*, pages 80–89. IEEE.

Granmo, O.-C., Glimsdal, S., Jiao, L., Goodwin, M., Omlin, C. W., and Berge, G. T. (2019). The convolutional tsetlin machine. *arXiv preprint:1905.09688*.

Granmo, O.-C. (2018). The Tsetlin machine - a game theoretic bandit driven approach to optimal pattern recognition with propositional logic. *arXiv preprint:1804.01508*.

Jacovi, A., Shalom, O. S., and Goldberg, Y. (2018). Understanding convolutional neural networks for text classification. In *Proceedings of EMNLP Workshop BlackboxNLP*.

Joulin, A., Grave, É., Bojanowski, P., and Mikolov, T. (2017). Bag of tricks for efficient text classification. In *Proceedings of EACL*, pages 427–431.

Kim, Y. (2014). Convolutional neural networks for sentence classification. In *Proceedings of EMNLP*.

Kowsari, K., Brown, D. E., Heidarysafa, M., Meimandi, K. J., Gerber, M. S., and Barnes, L. E. (2017a). Hdltex: Hierarchical deep learning for text classification. In *Proceedings of IEEE ICMLA*, pages 364–371.

Kowsari, K., Brown, D. E., Heidarysafa, M., Meimandi, K. J., Gerber, M. S., and Barnes, L. E. (2017b). Hdltex: Hierarchical deep learning for text classification. In *Proceedings of IEEE ICMLA*, pages 364–371.

Lage, I., Chen, E., He, J., Narayanan, M., Kim, B., Gershman, S., and Doshi-Velez, F. (2019). An evaluation of the human-interpretability of explanation. *CoRR*, abs/1902.00006.

Le, H. H. and Viviani, J.-L. (2018). Predicting bank failure: An improvement by implementing a machine-learning approach to classical financial ratios. *Research in International Business and Finance*, 44:16–25.

Lei, J., Rahman, T., Shafik, R., Wheeldon, A., Yakovlev, A., Granmo, O.-C., Kawsar, F., and Mathur, A. (2021). Low-power audio keyword spotting using tsetlin machines. *Journal of Low Power Electronics and Applications*, 11(2):18.

Li, B., Friedman, J., Olshen, R., and Stone, C. (1984). Classification and regression trees (cart). *Biometrics*, 40(3):358–361.

Lin, J., Tremblay-Taylor, G., Mou, G., You, D., and Lee, K. (2019a). Detecting fake news articles. In *Proceedings of 2019 IEEE International Conference on Big Data*.

Lin, J., Tremblay-Taylor, G., Mou, G., You, D., and Lee, K. (2019b). Detecting fake news articles. In *Proceedings of 2019 IEEE International Conference on Big Data*.

Lipton, Z. C. (2018). The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, 16(3):31–57.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Proceedings of NIPS*.

Nguyen, D. (2018). Comparing automatic and human evaluation of local explanations for text classification. In *Proceedings of NAACL-HLT (Long Papers)*, pages 1069–1078.

Pennebaker, J., Boyd, R. L., Jordan, K., and Blackburn, K. G. (2015). The development and psychometric properties of liwc2015. Technical report.

Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *Proceedings of EMNLP*, pages 1532–1543.

Ribeiro, M. T., Singh, S., and Guestrin, C. (2016). " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of ACM SIGKDD*.

Rubin, V. L., Conroy, N., and Chen, Y. (2015). Towards news verification: Deception detection methods for news discourse. In *Proceedings of the Rapid Screening Technologies, Deception Detection and Credibility Assessment Symposium, at the 48th An-*

*nual Hawaii International Conference on System Sciences*.

Saha, R., Granmo, O.-C., and Goodwin, M. (2020). Mining interpretable rules for sentiment and semantic relation analysis using tsetlin machines. In *Proceedings of International Conference on Innovative Techniques and Applications of Artificial Intelligence*.

Serrano, S. and Smith, N. A. (2019). Is attention interpretable? In *Proceedings of ACL*, pages 2931–2951.

Shu, K., Mahudeswaran, D., Wang, S., Lee, D., and Liu, H. (2020a). Fakenewsnet: A data repository with news content, social context, and spatiotemporal information for studying fake news on social media. *Big data*, 83:171–188.

Shu, K., Zheng, G., Li, Y., Mukherjee, S., Awadallah, A. H., Ruston, S. W., and Liu, H. (2020b). Leveraging multi-source weak social supervision for early detection of fake news. *arXiv preprint:2004.01732*.

Strumbelj, E. and Kononenko, I. (2010). An efficient explanation of individual classifications using game theory. *The Journal of Machine Learning Research*, 11:1–18.

Tsetlin, M. L. (1961). On behaviour of finite automata in random medium. *Avtomat. i Telemekh*, 22(10):1345–1354.

Ustun, B. and Rudin, C. (2016). Supersparse linear integer models for optimized medical scoring systems. *Machine Learning*, 102(3):349–391.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *Proceedings of NIPS*, pages 6000–6010.

Xiong, W., Ni'mah, I., Huesca, J. M., van Ipenburg, W., Veldsink, J., and Pechenizkiy, M. (2018). Looking deeper into deep learning model: attribution-based explanations of textcnn. In *Proceedings of NIPS 2018 Workshop on Challenges and Opportunities for AI in Financial Services: the Impact of Fairness, Explainability, Accuracy, and Privacy*.

Yadav, R., Jiao, L., Granmo, O.-C., and Goodwin, M. (2021a). Human-level interpretable learning for aspect-based sentiment analysis. In *Proceedings of AAAI*.

Yadav, R. K., Jiao, L., Granmo, O.-C., and Goodwin, M. (2021b). Distributed word representation in tsetlin machine. *arXiv preprint:2104.06901*.

Yadav, R. K., Jiao, L., Granmo, O.-C., and Goodwin, M. (2021c). Interpretability in word sense disambiguation using tsetlin machine. In *Proceedings of ICAART*, pages 402–409.

Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., and Hovy, E. (2016). Hierarchical attention networks for document classification. In *Proceedings of HLT-NAACL*.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R., and Le, Q. V. (2019a). Xlnet: Generalized autoregressive pretraining for language understanding. In *Proceedings of NIPS*.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R., and Le, Q. V. (2019b). Xlnet: Generalized autoregressive pretraining for language understanding. In *Proceedings of NIPS*.