Question Classification using Interpretable Tsetlin Machine

Dragoș Constantin Nicolae
dragosnicolae555@gmail.com
Research Institute for Artificial Intelligence "Mihai Drăgănescu"
Bucharest, Romania

ABSTRACT

Question Answering (QA) is one of the hottest research topics in Natural Language Processing (NLP) as well as Information Retrieval (IR). Among various domains of QA, Question Classification (QC) is a very important system that classifies a question based on the type of answer expected from it. Generalization is a very important factor in NLP specifically in QA and subsequently in QC. There are numerous models for the classification of types of questions. Despite its good performance, it lacks the interpretability that shows us how the model can classify the output. Hence, in this paper, we propose a Tsetlin Machine based QC task that shows the interpretability of the model yet retaining the state-of-the-art performance. Our model is validated by comparing it with other interpretable machine learning algorithms.

CCS CONCEPTS

• **Information systems** → *Question answering*; • **Computing methodologies** → *Rule learning*.

KEYWORDS

Tsetlin machine, Question Classification, and Interpretability

1 INTRODUCTION

QA is one of the most important task in NLP. It basically deals with extraction of the answer for a specific question asked in natural language [13],[7]. QA on a whole is a system rather than a specific task which consists of several components like question understanding, result representation, and answer extraction [16]. Among these components, question understanding is further subdivided into two steps: identifying the focus words in the question and classification of the question based on semantic type. However, QC is considered as the most fundamental component in NLP to deal with. It supports selecting the answer for the QA model which directly influences the performance. For simple understanding, let us consider a question "*What is the name of the president of USA*?". If QC is able to classify it as the question related to a person's name, the QA model only has to look into answers with the name rather than going through the whole text corpus. Hence, QC is always included as the component of QA.

Recent studies show that QC is mainly based on the text classification. Although QC seems similar to traditional text/sentiment classification, they are distinct to a greater extent. The question is mostly very short and has less lexicon-based information than sentiment text corpus. Therefore, it needs further analysis of the words to obtain higher classification accuracy. Therefore, the question should be enriched with additional syntactic and semantic knowledge [8],[12], replacing or extending the vocabulary thereby making the question more general. Since QC depends heavily on machine learning approaches [9], feature extraction plays a vital role in accomplishing the target accuracy. Feature extraction is done using various lexical, syntactic features and parts of speech. Most of the machine learning algorithms are powerful to obtain good accuracy with QC data [17]. However, there always exists a limitation of interpretation in the model. Decision Tree despite having somewhat interpretable when have a complex tree makes it slightly difficult for a human to extract the meaning out of it. Similarly, a very powerful tool called deep neural network having impressive performance is still criticized for being black-box in nature [3].

Interpretability is a huge topic of interest in recent machine learning domain. It is a challenging task to find a perfect balance between interpretability and accuracy. The traditional interpretable machine learning algorithm suffers the barrier of accuracy whereas black-box deep neural network has higher accuracy than others. Hence, to find a perfect balance of accuracy - interpretability factor, we make use of recently introduced paradigm called Tsetlin Machine. Tsetlin Machine is a pattern classification method that extracts information in propositional logic based on Tsetlin Automata [14]. It has shown promising results in various dataset and domain including image, numerical and text data as compared to SVMs, Decision Tree, Random Forests, Naive Bayes, Logistic Regression and Neural Networks [1, 2, 4, 5, 15]. Since, Tsetlin Machine works on bit manipulation, it is computationally efficient. This combination of accuracy, interpretability and computational simplicity makes this a promising tool in machine learning. In this paper, we make use of Tsetlin Machine for question classification on widely used UIUC QC dataset [10]. The model shows that it picks up very important features with minimum use of semantic information.

Rest of the paper are organized in following manner. Section 2 shows the information of dataset and required preprocessing. Section 3 explains the Tsetlin Machine architecture and results is explained in Section 4. At last the paper is concluded in section 5.

2 PROPOSED MODEL

2.1 Dataset and Preprocessing

Among many classification standards for the task of question categorization, UIUC hierarchical classification standard is the most used one. This dataset consists of questions that are divided into two layers. The first layer consists of 6 coarse categories and the second layer consists of 50 named as fine categories. We will consider 6 class system for our model which is shown in Table 1. Here, these six classes are the types of answers that each questions hold in the dataset where ABBR is abbreviation, DESC is description, ENTY is entity, HUM is human, LOC is location, and NUM is numberical. The dataset has 6000 questions from different sources to build UIUC

Table 1: Sample of UIUC dataset.

Questions	Class
What is the full form of .com ?	ABBR
What are liver enzymes ?	DESC
What 's the term for a young fox ?	ENTY
Who killed Gandhi ?	HUM
Where is the Loop ?	LOC
When did beethoven die ?	NUM

which is splited into training data (5500 questions) and testing data (500 questions).

Feature extraction is the most important task of the classification. Any model is highly dependent on how the features are extracted. Datasets like sentiment analysis and IMDB reviews are highly enriched with vocabulary and repeated words which makes the whole set more general to find the pattern for classification. However, the UIUC dataset included questions that are shorter in length and lack sufficient vocabulary and semantic knowledge. Hence, there are many preprocessing methods to enrich this data with additional information like parts of speech tagging, finding common hypernyms, name entity recognition, and using various embedding like word2vec, context2vec, Elmo and BERT [11]. Despite the availability of such a pre-trained knowledge-based approach, we aim to use minimal knowledge from external sources and utilize most of the information in the questions.

In general, the types of expected answers are highly dependent on the interrogative words in the question like what, which, when, where, who, how and why. The rest of the words in the question supports the interrogative words for proper semantic knowledge. However, these secondary words are not as important as interrogative words. So, we will consider two types of features in a question: 1). Primary: Interrogative words and 2). Secondary: context words. In this dataset, we have a high occurrence of interrogative words that can be used as the general features. These words alone can categorize the question to a particular type. For example, How many .....? This question clearly states that the expected answer is a numerical value categorizing it to NUM. However, questions like What is .....? can swing in any direction like What is Science? can be description and What is the distance between London to Dublin? can be a numerical value. Hence, we need to consider the context words as well. This brings us to another problem of non-general context in each question. It means the context words are not repetitive and considering all of them makes the vocabulary size too large and input vector too sparse. Hence, we try to make each question more general to reduce the vocabulary size. For this, we use Name Entity Recognition (NER) to extract whatever extra general information we can get.

To reduce the vocabulary size, the dataset is scanned to obtain the most common words. As expected the most repeated words are the interrogative words and word 'the' which appears in almost all the sentences [*the* : 3900, *what* : 3589, *how* : 797, *who* : 662, *name* : 406....., *where* : 301, *when* : 158]. Hence, we use NER from the spacy library [6] to generalize each question with its name entity. Each question is tokenized and appended with its NER tag along with it as [Who killed Gandhi ? PERSON]. This addition of NER tag increases the generalization in the dataset which makes most common words in vocabulary as [*the* : 3900, *what* : 3589, *PERSON* : 1295, *GPE* : 1011, ..., *How* : 797, *ORG* : 792]. After this preprocessing the most common words that are repeated at least 8 times are taken as the vocabulary list and input feature is prepared as the bag-of-words.

2.2 System Architecture

Tsetlin Machine solves the classification problem by pattern matching where a class is represented by several sub-pattern each pointing to certain features to differentiate values [14]. The Tsetlin Machine designs a model that presents sub-patterns in an effective way yet maintaining very simple management. Basically, Tsetlin Machine represents a particular class with the series of clauses where each clause is composed of sub-patterns by means of the conjunction of literals. Literals are the propositional variables or their negation. Each of these literals take the value 0 or 1.

Let us assume a feature vector $[x_1, x_2, x_3, ..., x_n]$ which is the bag of words for the preprocessed input that has n propositional variables (length of vocabulary) and x_k each taking value either 0 or 1. If there are z classes and l sub-patterns are required to represent a class, then the pattern classification problem can be captured by $z \times l$ number of conjunctive clauses C_i^j, $1 \leq j \leq z$, $1 \leq i \leq l$. The output of the classifier, y^j, $1 \leq j \leq z$ is given by:

$$C_i^j = 1 \wedge \left(\bigwedge_{k \in K_i^j} x_k \right) \wedge \left(\bigwedge_{k \in \bar{K}_i^j} \neg x_k \right).$$

$$y^j = \Sigma_{i=1}^{l} C_i^j.$$

Here, K_i^j and $\bar{K}_i^j$ are the non-overlapping subsets of the input variable indexes. The subsets here decides which of the propositional variables take part in the clause and whether they will be negated or not.

The Tsetlin Machine takes n propositional variables as input. For each variable, there are two literals formed, the original literal (x_k) itself and its negation ($\neg x_k$). For each clause C_i^j, every literals in it is assigned with unique Tsetlin Automata (TA) that decides whether to include or exclude its assigned literal in the given clause. Here for n input variables, we need $2n$ TA. This TA team construct a conjunction of the literals that TA selects to be included as shown in Fig. 1. When the all the included literals evaluate to 1, conjunction outputs 1 otherwise 0. Now, the next step is the clauses and its role. The Tsetlin Machine consists of l clauses, each of them are associated with TA team. The number of clauses that are needed to a particular class depends on the number of sub-patterns associated with that class. Each clause casts a vote so that l clauses combined decide the output of the Tsetlin Machine. Here the clauses with odd indexes are denoted with positive polarity (+) and the clauses with even indexes are assigned negative polarity (-). Among these sub division, the clauses that has the positive polarity cast their vote in favor of the decision that input belongs to that class. On the other hand, clauses with negative polarity vote for the absence of that class and belonging to other classes. After the clauses have produced their output, a summation operator associated with the

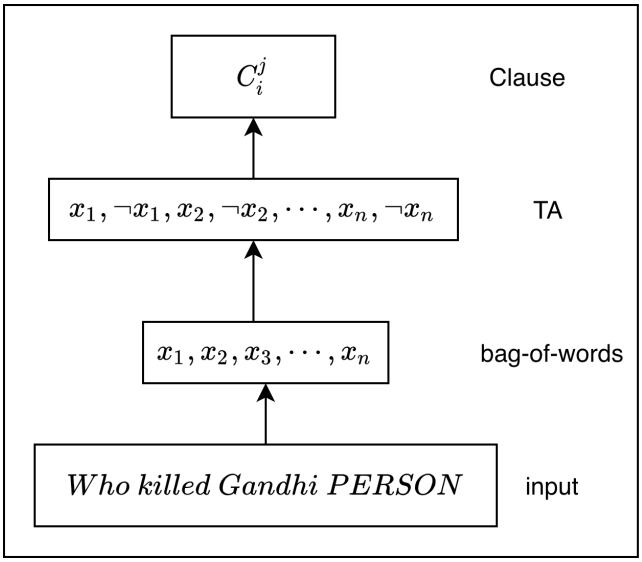

Figure 1: The basic Tsetlin Automata (TA) team representing a clause.

output class y^i sums the votes that it has received from the clauses which includes both positive and negative polarity. Finally, the output is obtained from following equations. If the summation is larger than or equal to zero, then the output y^i is produced as 1 as shown in Eq. 1 and 2.

$$f_\Sigma(X) = \left(\Sigma_{j=1,3,\ldots}^{l-1} C_j(X)\right) - \left(\Sigma_{j=2,4,\ldots}^{l} C_j(X)\right). \tag{1}$$

$$y = \begin{cases} 1 & \text{if } f_\Sigma(X) > 0, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

For multiclass Tsetlin Machine, multiple Tsetlin Machine is used as shown in Fig. 2. The final output for multiclass is given by the argmax operator having the highest number of vote sum . Since, the learning process is quite descriptive a Game for learning the Conjunctive clauses which can be seen in [4].

3 INTERPRETABILITY AND RESULTS

Here we present the performance of Tsetlin Machine on the UIUC dataset and the way we can interpret the learning process. Let's take step by step procedure to understand the interpretability of the learning process. We first initialize the necessary hyper-parameters (number of clauses = 2000, Threshold T = 50, specificity s = 20). These hyperparameters are tuned by grid search as done in finding the number of neurons in deep learning. Since there are 6 output classes, each class is assigned 2000 clauses to learn the sub-patterns. So there are altogether 12,000 clauses representing all the 6 classes. When a test data is passed to those 12,000 clauses, some of the clauses are triggered and the class that has the highest number of clauses giving the value 1 is decided to be the predicted class. In brevity, we can say that each class has a collection of sub-patterns.

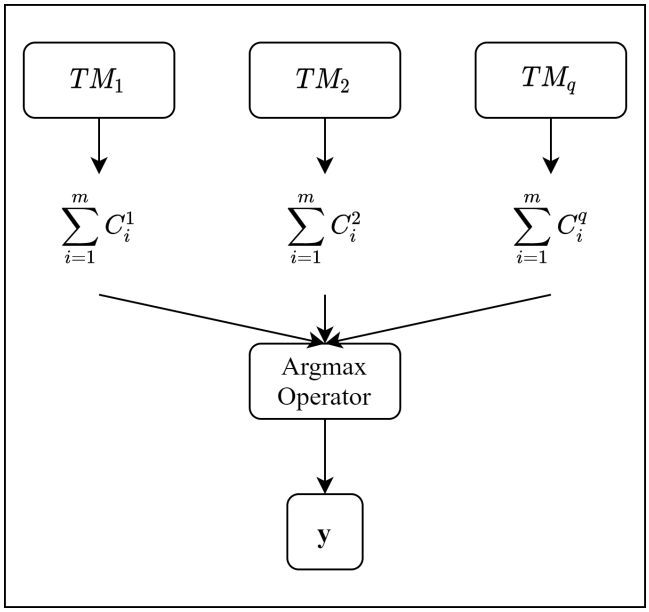

Figure 2: TA multiclass Tsetlin Machine [14].

C_1	Who	kill	not LOC	not ORG	not Country
C_2	Who	not ORDINAL	not LOC	not What	not When
C_3	Who	not What	not Where	not How	not Country
C_4	What	not kill	country	not GPE	LOC
⋮	⋮				⋮
C_{2000}	ORDINAL	not PERSON	not LOC	not Who	not Country

Figure 3: Clauses for class HUM with its sub-patterns.

C_1	Who	kill	not LOC	not ORG	not Country
C_2	Who	not ORDINAL	not LOC	not What	not When
C_3	Who	not What	not Where	not How	not Country
C_4	What	not kill	country	not GPE	LOC
⋮	⋮				⋮
C_{2000}	ORDINAL	not PERSON	not LOC	not Who	not Country

Figure 4: Clauses for class LOC with its sub-patterns.

When the test data satisfy with one of the lists of sub-pattern, it is denoted with that class.

Let us take an example to show how we can interpret the sub-pattern. For simplicity, lets consider only two classes, HUM and LOC. Now there are 200 clauses formed for each of these classes. The sub-patterns formed for class HUM and LOC is shown in Fig. 3 and 4. Now let us consider a test input for class HUM

$$S_1 = [Who\ killed\ Gadhi?].$$

After the necessary preprocessing and NER, the input becomes

$$S_1(NER) = ['Who',\ 'kill',\ 'Gadhi',\ 'PERSON'].$$

Similarly, the input that is associated with class LOC is

$$S_2(NER) = ['What',\ 'country',\ 'do',\ 'LOC'].$$

Now, if we pass $S_1(NER)$ to both clauses it satisfies 3 clauses C_1, C_2, C_3 and others for the sub-patterns of HUM whereas it only satisfies 1 clause C_3 for the sub-patterns of LOC. Hence, the summation of votes is higher towards HUM assigning it to class HUM. Similarly, $S_2(NER)$ satisfies only 1 clause C_4 in HUM compared to 3 clauses in LOC C_3, C_4, C_{2000} assigning it to class LOC.

As we can see from these clauses, it is quite easier to extract which inputs are included as original or negated to represent each class. The Tsetlin Machine has three parameters needed to be defined by the user and fine tuned as required. Here in this case we use 2000 clauses (C), threshold (T) of 80 and specificity (s) of 15. The parameter s is responsible for avoiding over-fitting as well as deciding how many literals can be included and excluded from the clause. We have manually assigned these parameters as the main aim is to demonstrate the interpretability of the model rather than the performance. The performance can be further enhanced by fine tuning those parameters. The Figure 5 shows the training and testing accuracy after each epoch. There are a total of 500 epochs to get a stable result on the Tsetlin machine model. This step-wise performance shows that the model immediately starts to learn the pattern more than 80% in less than 20 epochs. After 500 epochs the training accuracy reached around 94% and remains stable. Similarly, the test accuracy immediately reaches 80% in less than 15 epochs showing the fast convergence of the model. After 100 epochs, test accuracy reaches a stable 85% at most of the epochs. However, once the epoch crosses 200, the test accuracy is increased to a maximum of 87.2% giving the model an average accuracy of more than 86%.

This fascinating human interpretation does not come at any cost of accuracy. The performance can be seen when compared to other interpretable machine learning models. In this paper, we do not target the state-of-the-art result. However, we would like to represent the impact of same preprocessing on various machine learning models as shown in Table. 2. Here, we can see Tsetlin Machine has better performance than Support Vector Machine (SVM), Logistics Regression (LR), KNN, Decision Tree (DT) and Naive Bayes (NB). These models are more interpretable than the deep neural network and hence we assume these models to be the baseline for our comparison. Also further different preprocessing of input can enhance the accuracy as shown in [17]. However, we only focus on the interpretation of the QC model using TM thereby surpassing other machine learning algorithms. Although the training accuracy of DT is better than the rest of the model, its testing accuracy is not up to the mark with other models which shows the problem of over fitting. Tsetlin Machine performs significantly better than these models in terms of testing accuracy. However, SVM shows quite competitive performance with 85.2% test accuracy as compared to the Tsetlin machine's 87.2%. In brevity, we can say that such a simple human level interpretation of the model with reasonable accuracy certainly shows promising evidence of the Tsetlin machine in various applications.

4 CONCLUSION

QC, being an important part of QA, we presented a new paradigm called Tsetlin Machine for question classification based on the expected answer in UIUC dataset. At first, we used very basically preprocessing as removing unnecessary punctuation and then tagged each question with NER using the spacy library of python. This processed dataset is then scanned for most frequent words. Since the type of expected answers is highly dependent on interrogative words, almost all the interrogative words were included in the most frequent ones. However, some cases are very difficult to be distinguished with interrogative words only, hence NER added more generalization in the dataset. We then used the Tsetlin machine for the classification of 6 types of questions. The main aim of using the Tsetlin machine is its 3 important factors, accuracy, complexity, and interpretability. The accuracy obtained from it is compared to various traditional machine learning algorithms that show the Tsetlin machine to be superior with a human level interpretation of the model. Being a new model in machine learning, the Tsetlin machine is achieving either the same or high accuracy compared to the other interpretable machine learning models which makes it a promising tool in the field of NLP.

REFERENCES

[1] K. Darshana Abeyrathna, Ole-Christoffer Granmo, Xuan Zhang, Lei Jiao, and Morten Goodwin. 2019. The regression Tsetlin machine: a novel approach to interpretable nonlinear regression. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 378 (2019). Issue 2164.

[2] G. T. Berge, O. Granmo, T. O. Tveit, M. Goodwin, L. Jiao, and B. V. Matheussen. 2019. Using the Tsetlin Machine to Learn Human-Interpretable Rules for High-Accuracy Text Categorization With Medical Applications. *IEEE Access* 7 (2019), 115134–115146. https://doi.org/10.1109/ACCESS.2019.2935416

[3] Vanessa Buhrmester, David Münch, and Michael Arens. 2019. Analysis of Explainers of Black Box Deep Neural Networks for Computer Vision: A Survey. *ArXiv* abs/1911.12116 (2019).

[4] Ole-Christoffer Granmo. 2018. The Tsetlin Machine - A Game Theoretic Bandit Driven Approach to Optimal Pattern Recognition with Propositional Logic. *CoRR* abs/1804.01508 (2018). arXiv:1804.01508 http://arxiv.org/abs/1804.01508

[5] Ole-Christoffer Granmo, Sondre Glimsdal, Lei Jiao, Morten Goodwin, Christian W. Omlin, and Geir Thore Berge. 2019. The Convolutional Tsetlin Machine. arXiv:1905.09688 [cs.LG]

[6] Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. (2017). To appear.

[7] Eduard H. Hovy, Ulf Hermjakob, Chin-Yew Lin, and Deepak Ravichandran. 2002. Using Knowledge to Facilitate Factoid Answer Pinpointing. In *COLING*.

[8] Stanford University lere. 2010. High-Performance Question Classification Using Semantic Features.

[9] Xin Li, Xuanjing Huang, and Lide Wu. 2005. Question Classification using Multiple Classifiers. In *ALR/ALRN@IJCNLP*.

[10] Xin Li and D. Roth. 2002. Learning Question Classifiers. In *COLING*.

[11] Haozheng Luo, N. Liu, and Charles Feng. 2019. Question Classification with Deep Contextualized Transformer. *ArXiv* abs/1910.10492 (2019).

[12] Domen Marincic, Tomaz Kompara, and Matja Gams. 2012. Question Classification with Active Learning. In *TSD*.

[13] Dan I. Moldovan, Marius Pasca, Sanda M. Harabagiu, and Mihai Surdeanu. 2002. Performance Issues and Error Analysis in an Open-Domain Question Answering System. In *ACL 2002*.

[14] M. L. Tsetlin. 1961. paper On behaviour of finite automata in random medium. *Journal of Avtomatika. i Telemekhanika* 22 (1961), 1345 – 1354. http://mi.mathnet.

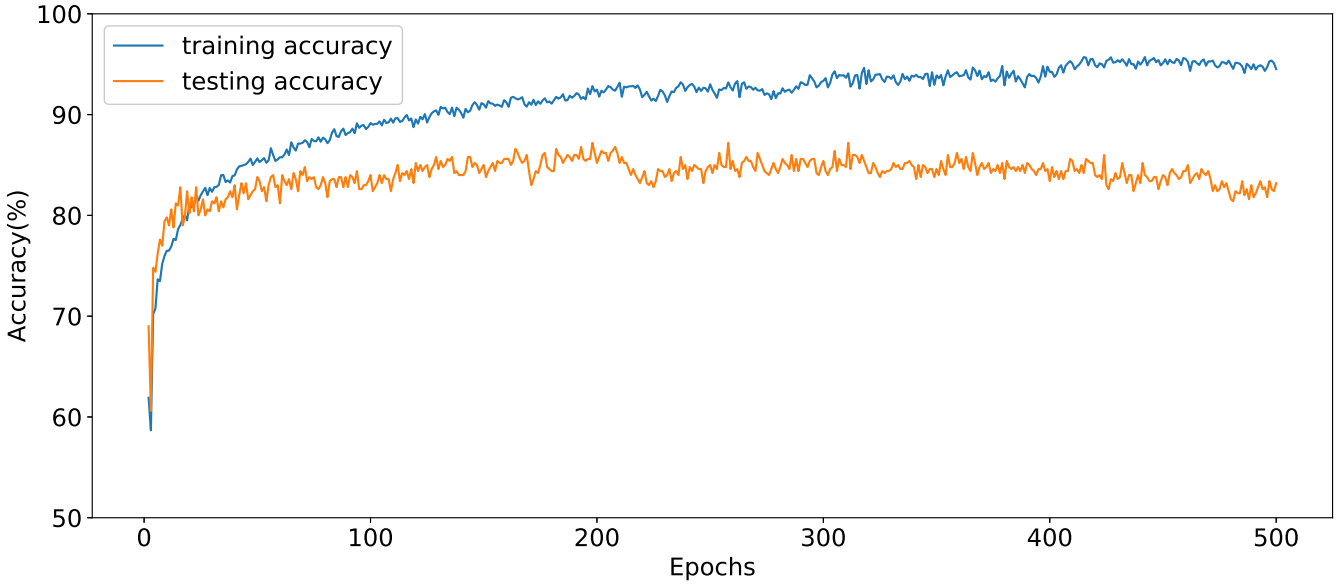

Figure 5: Training and testing accuracy after each learning epoch.

Table 2: Performance comparison with various Machine Learning algorithms..

ML algorithms	Training Accuracy (%)	Testing Accuracy (%)
SVM	92.11	85.20
LR	89.10	85.0
KNN	79.4	73.0
DT	99.0	81.40
NB	63.72	45.20
Tsetlin Machine	95.70	87.20

ru/at12417

[15] Rohan Kumar Yadav, Lei Jiao, Ole-Christoffer Granmo, and Morten Goodwin. 2021. Human-Level Interpretable Learning for Aspect-Based Sentiment Analysis. In *The Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI-21)*. AAAI.

[16] Li Ying-wei, Yu Zheng-tao, Meng Xiang-yan, Che Wen-gang, and Mao Cun-li. 2008. Question Classification Based on Incremental Modified Bayes. *2008 Second International Conference on Future Generation Communication and Networking* 2 (2008), 149–152.

[17] Chunting Zhou, Chonglin Sun, Zhiyuan Liu, and F. C. M. Lau. 2015. A C-LSTM Neural Network for Text Classification. *ArXiv* abs/1511.08630 (2015).