Attention enhanced capsule network for text classification by encoding syntactic dependency trees with graph convolutional neural network

Xudong Jia[1], Li Wang[1]

[1] College of data science, Taiyuan University of Technology, Taiyuan, Shanxi, China

Corresponding Author:
Li Wang [1]
79 Yingze West Street, Taiyuan, Shanxi, 030000, China
Email address: wangli@tyut.edu.cn

41 # Attention enhanced capsule network for text
42 # classification by encoding syntactic dependency trees
43 # with graph convolutional neural network

44

45 ## Abstract

46 Text classification is a fundamental task in many applications such as topic labeling, sentiment
47 analysis, and spam detection. The text syntactic relationship and word sequence are important
48 and useful for text classification. How to model and incorporate them to improve performance is
49 one key challenge. Inspired by human behavior in understanding text. In this paper, we combine
50 the syntactic relationship, sequence structure, and semantics for text representation, and propose
51 an attention-enhanced capsule network-based text classification model. Specifically, we use
52 graph convolutional neural networks to encode syntactic dependency trees, build multi-head
53 attention to encode dependencies relationship in text sequence, merge with semantic information
54 by capsule network at last. Extensive experiments on five datasets demonstrate that our approach
55 can effectively improve the performance of text classification compared with state-of-the-art
56 methods. The result also shows capsule network, graph convolutional neural network, and multi-
57 headed attention has integration effects on text classification tasks.

58

59 ## Introduction

60 Text classification is the basic task of text analysis with broad applications in topic labeling,
61 sentiment analysis, and spam detection. Text representation is one important way for
62 classification. From the analysis of text structure, the text is a sequence of words by certain rules.
63 Sequences of different word orders show different meanings. The sequence structure of a text
64 contains important information about the semantics of the text. Moreover, words in a text are
65 contextual, the distance dependence between words in the sequence structure will also affect the
66 meaning of the text. From the analysis of text composition, the text is composed of words or
67 phrases with different syntactic functions according to certain syntactic relations. For humans,
68 syntactic relations are the basis for editing and reading texts. According to the syntactic relation,
69 we can understand the subject, predicate, and object of the text to promptly understand the
70 semantics of the text. Whether it is text representation or text classification by text
71 representation, it is the key and important question for our research to extract information from
72 the sequence structure and syntactic relations of text.

73 Traditional methods represent text with handcrafted features, such as bag-of-words
74 (Joachims 1998; Mccallum and Nigam 1998), N-grams (Lin and Hovy 2003), and TF-IDF
75 (Zhang, Yoshida, and Tang 2008), and then adopted machine learning algorithms such as Naive
76 Bayes (Mccallum and Nigam 1998), logistic regression (Genkin and Madigan 2007), support
77 vector machine (Joachims 1998), etc. for classification. These methods ignore the word order
78 and semantic information, which has an important role in understanding the semantics of the text
79 (Pang, Lee, and Vaithyanathan 2002). On the contrary, other methods use word2vec and glove

80   (Mikolov and Sutskever 2013; Penning- ton, Socher, and Manning 2014) to represent text that
81   can represent the semantic information of words, and then adopted convolutional neural network
82   (CNN) (Kim 2014; Zhang, Zhao, and Lecun 2015; Conneau et al. 2017), long short-term
83   memory networks (LSTM) (Mousa and Schuller 2017), Capsule networks (CapsNet) (Zhao et al.
84   2018) and other deep neural networks for text classification. These methods can effectively
85   encode the word order and semantic information. Kim et al. (Kim 2014) proposed a CNN model
86   that adopted convolution filters to extract local text semantic features for text classification,
87   leading to the model loss of a part of location information (Zhao et al. 2018). Capsule networks
88   with vector neural units and dynamic routing algorithms can effectively overcome the
89   disadvantages of CNN (Sabour, Frosst, and Hinton 2017; Xi, Bing, and Jin 2017). Zhao et al.
90   (Zhao et al. 2018) optimized the capsule networks for text classification (Capsule-A), and further
91   experiments show the effectiveness of capsule networks for text classification. The
92   characteristics of capsule networks are our starting point. However, this model has limitations in
93   recognizing text with semantic transitions since it cannot encode long-distance dependencies and
94   the global topology, which affects the effectiveness of complex text classification. The attention-
95   based approach is effective in overcoming the problem of distance dependencies of sequence
96   structure. Both Transformer (Vaswani, Shazeer, and Parmar 2017) and Bidirectional Encoder
97   Representations from Transformers (BERT) (Alaparthi, and Mishra 2020) use multi-head
98   attention as a basic unit to extract text features. It's part of our motivation that it can extract the
99   distance dependencies information of the text from different subspaces.
100   Syntactic information is a different attribute for text. It describes the syntactic dependency
101   relationship between words in a sentence. It can be represented as the syntactic dependency tree
102   and show the global topology. Figure 1 is a syntactic dependency tree, where 'monkey' is the
103   subject of the predicate 'eats', and 'apple' is its object. GCN is a graph convolutional network
104   that operates on graphs and induces embeddings of nodes based on the properties of their
105   neighborhoods. It can capture the information of the immediate neighbors of nodes at most K
106   hops away (Marcheggiani and Titov 2017; Duvenaud and Maclaurin 2015). Marcheggiani et al.
107   (Marcheggiani and Titov 2017) used GCN to encode syntactic dependency trees to generate
108   word representations, and combined with long short-term memory networks for semantic role
109   labeling tasks. These works show that GCN can effectively extract syntactic information in
110   syntactic dependency trees. This is another motivation for us.
111   The text syntactic relationship and word sequence are important and useful for text
112   classification. How to model and incorporate them to improve performance is one key challenge.
113   In this paper, we combine the syntactic relationship, sequence structure, and semantics for text
114   representation, and propose a novel model that utilizes GCN for syntactic relationship, multi-
115   head attention for words, and corporate them in capsule network for text classification.
116   The contributions of this paper can be summarized as follows:
117   •   We incorporate syntactic relationship, sequence structure, and semantics for text
118   representation.

119 • We introduce GCN to extract syntactic information for dependencies relationship
120   representation.
121 • We build multi-head attention to encode the different influences of words to enhance the
122   effect of capsule networks on text classification.
123 • We show that CapsNet, GCN, and multi-head attention have an integration effect for text
124   classification.
125

Related work
127 **Machine learning-based methods**
128 Early methods adopted the typical features such as bag-of-words (Joachims 1998; Mccallum and
129 Nigam 1998), N-grams (Lin and Hovy 2003), and TF-IDF (Zhang, Yoshida, and Tang 2008)
130 features as input and utilized machine learning algorithms such as support vector achine (SVM)
131 (Joachims 1998), logistic regression (Genkin and Madigan 2007), naive Bayes (NB) (Mccallum
132 and Nigam 1998) for classification. However, these methods ignore the text word order and
133 semantic information, and usually heavily rely on laborious feature engineering.
134

135 **Deep learning-based methods**
136 With the introduction of distributed word vector representation (Word Embedding) (Mikolov and
137 Sutskever 2013; Pennington, Socher, and Manning 2014), neural networks-based methods have
138 substantially improved the performance of text classification tasks by encoding text semantics.
139   CNN was first applied to image processing. Kim et al. (Kim 2014) proposed the CNN-
140 based text classification model (TextCNN). The model uses convolution filters to extract local
141 semantic features and improved upon the state of the art on 4 out of 7 tasks. Zhang X et al.
142 (Zhang, Zhao, and Lecun 2015) proposed the character-level CNN model, which extracts
143 semantic information from character-level original signals for text classification tasks. Conneau
144 et al. (Conneau et al. 2017) proposed very deep convolutional networks to learn the hierarchical
145 representation for text classification. Being a spatially sensitive model, CNN pays a price for the
146 inefficiency of replicating feature detectors on a grid.
147   Recently, Hinton et al. (Sabour, Frosst, and Hinton 2017) proposed the CapsNet model,
148 which uses vector neural units and dynamic routing update mechanisms, and verified its
149 superiority in image classification. Zhao et al. (Zhao et al. 2018) proposed the text classification
150 model based on CapsNet (Capsule-A), which adopted CapsNet to encode text semantics, and
151 proved that its classification effect is superior to CNN and LSTM. In CapsNet, the feature is
152 represented by a capsule vector instead of a scalar (activation value output by neuron). Different
153 dimensions in a vector can represent different properties of a feature. For a text feature, it often
154 means different meanings in different semantic relations. We use capsules to represent text
155 features to learn the semantic information of different dimensions of text features. On the other
156 hand, the similarity between features at different levels is different. For building a high-level
157 feature, lower levels with high similarity have a higher weight. CapsNet can learn this similarity

158 relationship through a dynamic routing algorithm. Although CapsNet can effectively improve
159 coding efficiency, it still has limitations in recognizing text with semantic transitions.
160   Attention mechanisms are widely used in tasks such as machine translation (Vaswani,
161 Shazeer, and Parmar 2017) and speech recognition (Chorowski et al. 2014). Lin et al. (Lin et al.
162 2017) proposed the self-attention mechanism that can encode long-range dependencies. Vaswani
163 et al. (Vaswani, Shazeer, and Parmar 2017) proposed a machine translation model (Transformer)
164 based on multi-head attention. Alaparthi et al. (Alaparthi, and Mishra 2020) proposed a pre-
165 trained model of language representation (BERT) that also takes multi-head attention as its basic
166 component. The basic unit of multi-head attention is scaled dot-product attention. Multi-head
167 attention allows the model to jointly attend to information from different representation
168 subspaces at different positions. Attention is to extract the long-distance dependencies in the text
169 by calculating the similarities between words in the text. The words in the text can express
170 different meanings in different semantic scenarios. The representation of words is different in
171 different semantic spaces. Similar to ensemble learning, multi-head attention can put text in
172 different semantic spaces to calculate attention and get integrated attention. The location
173 information cannot be obtained by relying solely on the attention mechanism, and location
174 information also has an important influence on understanding text semantics. Kim et al. (Kim,
175 Lee, and Jung 2018) proposed a text sentiment classification model combining attention and
176 CNN, but it is still limited by the disadvantages of CNN. Although BERT has been particularly
177 effective on many tasks. But it requires a lot of data and computing resources for pre-training.
178 Therefore, our research is still valuable.

179 **Syntactic information-based Methods**
180 CNN, RNN, and most deep learning-based methods always utilized word local topology to
181 represent text. Word order and semantic, text syntactic information all have important influences
182 on text classification. Some researchers have done some work on text syntactic information for
183 different tasks. A text is made up of words that represent different syntactic elements, such as
184 subject, predicate, object, and so on. Different syntactic elements are interdependent. A syntax
185 dependency tree is a kind of tree structure, which describes the dependency relationship between
186 words. Figure 1 shows an example of a syntax dependency tree. There are many tools (like
187 StanfordNLP) to generate syntactic dependency trees by analyzing syntactic dependency
188 relations.
189   Eriguchi et al. (Eriguchi, Tsuruoka, and Cho 2017) adopted RNN to integrate syntactic
190 information for machine translation. Le et al. (Le and Zuidema 2014) introduced RNN to model
191 syntax dependency tree. Miwa et al. (Eriguchi, Tsuruoka, and Cho 2017) used sequential LSTM
192 and tree-LSTM to extract syntactic relations.
193   The syntactic dependency tree is also a kind of graph data. Bastings et al. (Bastings et al.
194 2017) used GCN to encode the syntactic dependency tree and combined it with the CNN for
195 machine translation. Marcheggiani et al. (Marcheggiani and Titov 2017) used GCN to encode
196 syntactic dependency trees to generate word representations and combined it with LSTM for the
197 role labeling task. These works show that GCN can effectively extract syntactic information in

198   syntactic dependency trees. The syntactic relationship is presented as a tree structure. A tree is
199   also a form of a graph. Secondly, the sequence structure-based model represents the syntactic
200   tree as a sequence according to some rules of nodes. But the sequence is directional, and nodes in
201   a tree are not sequential. GCN can directly consider the relationship between nodes in a syntax
202   tree.
203   In summary, we aim to propose a novel model named Syntax-AT-CapsNet that uses multi-
204   head attention to extract long-distance dependencies information and that uses GCN to encode
205   syntactic dependency trees to extract syntactic information, which enhances the effect of capsule
206   networks on text classification tasks.
207

Syntax-AT-CapsNet model

209   Our Syntax-AT-CapsNet model consists of the following three modules as depicted in (Fig. 2).
210   •   Attention module. It is composed of an attention layer that adopts multi-head attention. It
211   encodes the dependency relationship between words in the text sequence and important
212   word information to form a text representation.
213   •   Syntax module. It is composed of GCN. It encodes the syntax dependency tree, extracts
214   the syntactic information in the text to for a text representation.
215   •   Capsule network module. It is a capsule network with 5-layer. Based on the text
216   representation output by the Attention module and the Syntax module, it further extracts text
217   semantic and structural information to classify the text.
218

Input

220   The input of the Syntax-AT-CapsNet model is defined as the sentence matrix $\boldsymbol{X}$:

$$X = [x_1, x_2,..., x_L] \in \mathbb{R}^{L \times d}, \tag{1}$$

222   where $x_i \in \mathbb{R}^d$ is the word vector of the i-th word in the sentence, L is the length of the sentence,
223   and d is the embedding vector size of words.
224

Attention module

226   The Attention module is shown in (Fig. 3). The calculation of attention in this module can be
227   divided into five steps.
228   First step, linearly transform the input sentence matrix X and divide it into three matrices: $\boldsymbol{Q}$
229   $\in \mathbb{R}^{L \times d}$、$\boldsymbol{K} \in \mathbb{R}^{L \times d}$、$\boldsymbol{V} \in \mathbb{R}^{L \times d}$:

$$[Q,K,V] = spilt[X \cdot W], \tag{2}$$

231   where $\boldsymbol{W} \in \mathbb{R}^{d \times 3d}$ is the transform matrix, and $spilt$ denotes the division operation.
232   Second step, linearly projects the matrices $\boldsymbol{Q,K,V}$ onto h different linear subspaces:

$$[Q_1,...,Q_h] = \left[Q \cdot W^{Q_1},...,Q \cdot W^{Q_h}\right],$$

$$[K_1,...,K_h] = \left[K \cdot W^{K_1},...,K \cdot W^{K_h}\right],$$

$$[V_1,...,V_h] = \left[V \cdot W^{V_1},...,V \cdot W^{V_h}\right], \tag{3}$$

236 where $Q_i \in \mathbb{R}^{L \times \frac{d}{h}}, K_i \in \mathbb{R}^{L \times \frac{d}{h}}, V_i \in \mathbb{R}^{L \times \frac{d}{h}}$ is the mapping of Q,K,V on the i-th subspace, $W^{Q_i} \in$

237 $\mathbb{R}^{d \times \frac{d}{h}}, W^{K_i} \in \mathbb{R}^{d \times \frac{d}{h}}, W^{V_i} \in \mathbb{R}^{d \times \frac{d}{h}}$ is the transform matrix, $i = [1,...,h]$. The purpose of this step is

238 to compute multiple attention values in parallel. At the same time, the dimension of input matrix

239 is reduced to reduce the calculation pressure caused by multiple calculation.

240   Third step, calculate the attention on each subspace in parallel:

241 $$head_i = softmax\left(Q_i \cdot K_i^T/\sqrt{d}\right)V_i, \qquad (4)$$

242 where $head_i$ is the attention value on the i-th subspace, $softmax$ denotes the $softmax$

243 function (Vaswani, Shazeer, and Parmar 2017). In fact, Q_i and K_i represents the sentence matrix

244 on the subspace. It's divided by $\sqrt{d}$ in case the dot product gets too big. The weight of the

245 sentence matrix on the subspace V_i is obtained by calculating the dot product of Q_i and K_i and

246 using a $softmax$ function.

247   Fourth step, concat the attention values on each subspace and get the attention value of the

248 entire sentence through linear transformation:

249 $$Multi_{head} = concat(head_1,...,head_h)W^M, \qquad (5)$$

250 where $W^M \in \mathbb{R}^{d \times d}$ is the transform matrix, $Multi\_head$ is the attention value of the entire

251 sentence, and $concat$ denotes the concat operation.

252   The final step, Connect the attention value $Multi\_head$ to the original sentence matrix X

253 to get the sentence matrix output by the module:

254 $$X_1 = residual\_Connect(X,Multi\_head), \qquad (6)$$

255 where $X_1 \in \mathbb{R}^{L \times d}$ is the output of the attention module, and $residual\_Connect$ denotes the

256 residual connection operation.

257

258 **Syntax module**

259 The Syntax module is shown in (Fig. 4). The Syntax module uses GCN to encode syntactic

260 dependency trees, which can encode syntactic relationships between words in a text into word

261 vectors. The module first needs to use a natural language processing tool (we adopted

262 StanfordNLP) to generate the syntactic dependency tree of the input sentence, and construct its

263 adjacency matrix. The adjacency matrix construction algorithm in this paper is shown in

264 Algorithm 1. Since the syntactic relationship between word nodes in the syntactic dependency

265 tree has direction, when constructing the adjacency matrix, the syntactic dependency tree is used

266 as a directed graph. In addition, in order not to be disturbed by its word vector, the node is not

267 provided with a self-loop. As shown in (Fig. 5), the adjacency matrix corresponds to the example

268 sentence "The Monkey eats an apple" shown in (Fig. 1) and is generated by our method. The

269 input sentence matrix and adjacency matrix are further passed through the GCN to obtain a text

270 representation containing syntactic information.

271

272

273

274

Algorithm 1: Adjacency matrix construction algorithm

Input: sentence matrix X

Output: adjacency matrix adj

1 define zero matrix adj

2 Use the StanfordNLP tool to generate the syntax dependency tree $Tree$;

3 **for** each edge <i, j> in $Tree$ **do**:

4   Put the element $adj[i,j]$ in the matrix adj to 1;

5 **end for;**

6 **return** the adjacency matrix adj;

7 **end.**

275

276   The calculation in the Syntax module can be divided into two steps.

277   First step, use StanfordNLP tool to generate syntactic dependency tree and construct

278 adjacency matrix adj.

279   Second step, perform a two-layer graph convolution operation on the input sentence matrix

280 X and adjacency matrix adj:

$$G = relu((X \cdot adj)W^{t1}), \tag{7}$$

$$X_2 = softmax((G \cdot adj)W^{t2}), \tag{8}$$

283 where $G \in \mathbb{R}^{L \times d}$ is the output of the first layer graph convolution operation, $X_2 \in \mathbb{R}^{L \times d}$ is the

284 output of the second layer graph convolution operation, and $adj \in \mathbb{R}^{L \times L}$ is adjacent matrix, W^{t1}

285 、 $W^{t2} \in \mathbb{R}^{d \times d}$ are parameter matrices. In operation (7), the adjacency matrix and sentence

286 matrix go through the first level graph-convolution operation. The relationship between nodes

287 directly connected is obtained. Then, through operation (8), the relationship between nodes

288 indirectly connected nodes is calculated.

289

290 **Capsule network module**

291 The capsule network module is composed of a fusion layer, a convolution layer, a primary

292 capsule layer, a convolution capsule layer, and a fully connected capsule layer. It uses the text

293 representation output by the attention module and the syntax module as input to further extract

294 features. Each layer in the module can extract different levels of features. By further combining

295 low-level features to obtain higher-level features, and finally form a feature representation of the

296 entire text for classification.

297   The first layer is the fusion layer, put the text representations X_1 and X_2 output by the

298 syntax module and the attention module into a single layer network:

$$X_3 = W^{f1} \cdot X_1 + W^{f2}X_2, \tag{9}$$

300 where $W^{f1}, W^{f2} \in \mathbb{R}^L$. Two sentence matrices are combined by linear transformation through this

301 step.

302   The second layer is the convolutional layer, which extracts N-gram phrase features at
303 different positions in the text. This layer uses k_1 convolution filters to perform convolution on
304 the sentence matrix X_3 to obtain the N-gram feature matrix M:

305
$$M = \left[M_1, M_2, ..., M_{k_1}\right] \in \mathbb{R}^{(L - N + 1) \times k_1}, \tag{10}$$

306 where $M_{k_1} = [m_1, m_2, ..., m_{L - N + 1}] \in \mathbb{R}^{L - N + 1}$ is the k_1-th column vector in M, each element

307 m_i in this vector is obtained by operation (11):

308
$$m_i = f\left(W^{c1} \cdot x_{i:i + N - 1} + b_1\right), \tag{11}$$

309 where f denotes the nonlinear activation function, $W^{c1} \in \mathbb{R}^{N \times d}$ is the k_1-th convolution filter,
310 $x_{i:i + N - 1}$ denotes that N-word vectors in the sentence are connected in series, b_1 is bias item.
311 The original features are extracted by convolution in this step.
312   The third layer is the primary capsule layer, which combines the N-gram phrase features
313 extracted at the same location as capsules. This layer uses k_2 transformation matrices to
314 transform the feature matrix M into the primary capsule matrix P:

315
$$P = \left[P_1, P_2, ..., P_{k_2}\right] \in \mathbb{R}^{(L - N + 1) \times k_2 \times l}, \tag{12}$$

316 where $P_{k_2} = [p_1, p_2, ..., p_{L - N + 1}] \in \mathbb{R}^{(L - N + 1) \times l}$ is the k_2-th column capsule in P, each capsule

317 is obtained by operation (13):

318
$$p_i = g\left(W^{c2} \cdot M^i + b_2\right), \tag{13}$$

319 where g denotes the nonlinear compression function, $W^{c2} \in \mathbb{R}^{k_2 \times 1 \times l}$ is the k_1th transformation
320 matrix, M^i is the i-th row vector in M, and b_2 is bias item. The primary capsules are constructed
321 by linearly transforming the original features of the same location.
322   The fourth layer is the convolution capsule layer, which uses a shared transformation matrix
323 to extract local capsules, similar to the convolution layer. This layer uses k_3 transformation
324 matrices to perform capsule convolution operation on P to obtain the capsule matrix U:

325
$$U = \left[U_1, U_2, ..., U_{k_3}\right] \in \mathbb{R}^{(L - N - N_1 + 2) \times k_3 \times l}, \tag{14}$$

326 where $U_{k_3} = \left[u_1, u_2, ..., u_{L - N - N_1 + 2}\right] \in \mathbb{R}^{(L - n_2 + 1) \times l}$ is the k_3-th column capsule in U, each

327 capsule is obtained from the N_1 line capsules in P:

328
$$u_r \leftarrow P^{i:i + \left(N_1 \times k_2\right) - 1}, \tag{15}$$

329 where $P^{i:i + \left(N_1 \times k_2\right) - 1}$ represents N_1 rows of capsules, and each capsule is linearly converted to
330 obtain a prediction vector:

331
$$\hat{p}_i = W^{c3} \cdot p_i + \hat{b}_i, \tag{16}$$

332 where $W^{c3} \in \mathbb{R}^{l \times l}$ is the k_3-th conversion matrix, $\hat{b}_i$ is the offset term. It's the same thing as
333 convolution, except the basic units become capsules. And the prediction vector is operated (17):

334 $$\mathbf{u}_r = g\left(\Sigma_i c_i \cdot \hat{\boldsymbol{p}}_i\right), \tag{17}$$

335 where g is a nonlinear activation function, c_i is the coupling coefficient, which is updated with a

336 dynamic routing algorithm (Zhao et al. 2018). The similarity between the primary capsules and

337 the generated convolutional capsules is different within a window. A primary capsule with a high

338 similarity should be given a higher weight. Equation (17) is based on this principle.

339   The last layer is the fully connected capsule layer, which is used to form the capsule Y

340 representing the category:

341 $$Y = \left[\boldsymbol{y_1}, \boldsymbol{y_2}, ..., \boldsymbol{y_j}\right] \in \mathbb{R}^{j \times l}, \tag{18}$$

342 where $\boldsymbol{y_j} \in \mathbb{R}^l$ denotes the capsule of the j-th category. The capsules in U are linearly

343 transformed (16) to obtain the prediction vector $\boldsymbol{u}_{j|r}$, and the operation (17) is performed to

344 obtain $\boldsymbol{y_j}$. The fully connected capsule layer is shown in (Fig. 6), it can also represent a

345 convolution window operation at the fourth layer.

346   Finally, the modulus of the capsule vector representing the category in the fully connected

347 capsule layer is taken as the probability of belonging to the category.

348

349 **Syntax-AT-CapsNet learning algorithm**

350 The learning algorithm of Syntax-AT-CapsNet is shown in Algorithm 2. When the model is

351 learning, the coupling coefficients are updated by the dynamic routing algorithm, and the global

352 parameters of the model are updated by the back propagation algorithm. Then the trained

353 Syntax-AT-CapsNet model parameters can be obtained. During prediction, the classification

354 results can be obtained through the sequential calculation of each module in the model.

Algorithm 2: Syntax-AT-CapsNet learning algorithm

Input: sentence matrix X; training batch Epoch;

Output: Trained Syntax-AT-CapsNet model parameters;

1 Initialize the parameters in the model;

2 **for** each training batch **do**:

     //Attention module: Step 3-Step 7

3   Linearly transform and divide the sentence matrix X into $\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}$;

4   Project Q, K, V into h subspaces to obtain $\boldsymbol{Q}_i, \boldsymbol{K}_i, \boldsymbol{V}_i$;

5   Calculate attention on each subspace to get $\boldsymbol{head}_i$;

6   Concat each subspace $\boldsymbol{head}_i$ and linearly transform to get $\boldsymbol{Multi\_head}$;

7   Residuals connect $\boldsymbol{Multi\_head}$ and X as sentence matrix $\boldsymbol{X_1}$;

     //Syntax module: Step 8-Step 9

8   Construct adjacency matrix adj

9   Perform graph convolution operation on X and adj to get sentence matrix $\boldsymbol{X_2}$

     //Capsule network module: Step 10-Step 14

10   Put $\boldsymbol{X_1}$ and $\boldsymbol{X_2}$ into MLP get the sentence matrix $\boldsymbol{X_3}$

11   Perform convolution operation on $\boldsymbol{X_3}$ to obtain an N-gram matrix $\boldsymbol{M}$;

12  Convert the N-gram feature matrix M to the capsule matrix P;

13  Perform capsule convolution and routing on P to obtain the capsule matrix U;

14  Perform capsule calculation and routing on U to obtain category capsule Y;

15  Calculate loss and update parameters by back propagation;

16 **end for**;

17 **return** the trained Syntax-AT-CapsNet model parameters;

18 **end.**

355

356

357 **Experimental details**

358 We did extensive experiments to verify the effect of Syntax-AT-CapsNet model on the single
359 label and multi-label text classification tasks and designed more ablation experiments to
360 demonstrate the role of each module.

361

362 **Data sets**

363 We choose the following five datasets in our experiments: movie reviews (MR) (Miwa and
364 Bansal 2016), subjectivity dataset (Subj) (Pang and Lee 2004), customer review (CR) (Hu and
365 Liu 2004), Reuters-21578 (Lewis 1992).

366   The data in MR, Subj, and CR have two categories and are used for the single-label
367 classification tasks, where MR and Subj are composed of movie sentiment review data, CR is
368 composed of product reviews from Amazon and Cnet. The Reuters-21578 test set consists of
369 Reuters news documents. We selected 10788 news documents under the 8 category labels related
370 to economic and financial topics in Reuters-21578 and further divided them into two sub-
371 datasets (Reuters-Full and Reuters-Multi). In Reuters-Full, all texts are kept as the test set, and in
372 Reuters-Multi, only multi-label texts are kept as the test set. The experimental data description is
373 shown in Table 1.

374

375 **Evaluation index**

376 Exact Match Ratio (ER), Micro Averaged Precision (Precision), Micro Averaged Recall (Recall),
377 and Micro Averaged F1 (F1) were used as evaluation indexes in the experiment. Accuracy is
378 used instead of ER in the single-label classification.

379

380 **Parameter setting**

381 The experimental parameters of our work are as follows. In the model, input a 300-dimensional
382 word2vec word vector ($d = 300$). The attention module uses two heads of attention ($h = 2$).
383 The first layer of the capsule network module uses 32 convolution filters $\left(k_1 = 32\right)$, the window
384 size is 3 (N=3). The second layer uses 32 transformation matrices $\left(k_2 = 32\right)$ and 16-dimensional
385 capsule vectors ($l = 16$). The third layer uses 16 conversion matrices $\left(k_3 = 16\right)$, the window
386 size is 3 ($N_1 = 3$). The last layer uses 9 capsule vectors ($j = 9$) to represent 9 classes.

387   In model training, mini-batch with a size of 25 ($batch\_size$) are used, the training batch is
388 controlled to 20 ($Epoch = 20$), and the learning rate is set to 0.001 ($learning\_rate = 0.001$).
389 During the model test, for the single label classification task, the category label corresponding to
390 the capsule vector with the largest module length is taken. For the multi-label classification tasks,
391 the category labels corresponding to capsule vectors with a modulus length greater than 0.5 are
392 taken.
393

394 **Benchmark model**
395 In this paper, TextCNN, Capsule-A, and AT-CapsNet are used as benchmark models for
396 comparative experiments. TextCNN is a classic model of text classification based on CNN,
397 which is representative. Capsule-A is a text classification model based on capsule networks. AT-
398 CapsNet is a multi-headed attention capsule network text classification model.
399

Experimental results and discussion

401

Performance on single-classification and multi-classification tasks
403 The experimental result of single-label classification is shown in Table 2, and the multi-label
404 classification experiment result is shown in Table 3.
405   It can be observed from the experimental results:
406 •   Compared with the benchmark model (Table 2), our model achieved the best results of
407   accuracy, Recall, and F1 on the three binary classification data sets ER, Subj, CR.
408   Compared with AT-CapsNet (a multi-headed attention capsule network that does not
409   introduce syntax), it is found that the three data sets have a significant improvement, which
410   proves the value of introducing syntactic information in this article.
411 •   Compared with the benchmark models (Table 3), on the two multi-label data sets Reuters-
412   Full and Reuters-Multi, our model has achieved competitive results in four evaluation
413   indicators. And achieves the best results in ER, Recall, and F1, which indicates the
414   effectiveness of the model in the classification of multi-tag texts.
415   That is, our model has better effects on multi-label and single-label classification tasks than
416 benchmark methods. As described in the introduction, the problems with the baseline model are
417 the key to our research. This shows that our model overcomes the shortcomings of these models
418 to a certain extent. That's the purpose of our work.
419

Syntax module verification experiment
421 To show the effect of the syntax module, we did the following experiments and the experimental
422 results are shown in Table 4.
423   We can see that when the syntactic module is added to the benchmark model, the four
424 evaluation indicators have been significantly improved, which shows that the syntactic module
425 of this article can effectively improve the effect of text classification tasks. It also proves the

426 feasibility and value of extracting syntactic information with graph convolutional neural
427 networks. This also shows that we are right to learn from human reading behavior.
428
429 **Module ablation experiment**
430 The results of the module ablation experiment are shown in Table 5 and Table 6.
431   From the above experimental results, we can draw the following conclusions:
432 •   When controlling a single module (Table 5), the ablation of each module will cause the
433   classification effect to decrease to varying degrees, which shows that each module in our
434   model has a certain role in improving the text classification effect. In addition, by
435   comparing the reduced values, it was found that the capsule network had the greatest
436   influence, followed by attention, and the syntax module was smaller, which indicated the
437   correctness and value of taking the capsule network as the core module in this paper.
438 •   When two modules are controlled (Table 6), the ablation of any two modules will cause the
439   classification effect to decrease to varying degrees. Among them, the ablation syntax and
440   capsule network modules have the greatest influence, and the ablation syntax and attention
441   module have the second influence, and the ablation attention and the capsule network
442   module has the least influence, which shows that the syntactic module can function to the
443   greatest extent when combined with other modules. It also shows that the motivation of
444   using graph neural networks to encode syntactic information and other models is correct.
445 •   It can be seen from the above that when the attention module, the syntax module, or the
446   capsule network module in the model of this article is removed or partially removed, the
447   effectiveness of the model has declined to vary degrees. Since the syntactic module uses
448   graph convolutional neural networks, the above experiments also prove that graph
449   convolutional neural networks, capsule networks, and multi-head attention have an
450   integrated effect on text classification tasks. This also shows that we are on the right track
451   in building the model.

Conclusions

453 This paper proposes an enhanced capsule network text classification model Syntax-AT-CapsNet
454 for text classification tasks. The model first uses graph convolutional neural networks as
455 submodules to encode syntactic dependency trees, extract syntactic information in text, and
456 further integrate with sequence information and dependency relationships, thereby improving the
457 effect of text classification. Through model classification effect verification experiment, syntax
458 module verification experiment, and module ablation experiment, the effect of the model in this
459 paper on text classification and multi-label text classification task is verified, the function of
460 syntax module is demonstrated, and the integrated effect of graph convolutional neural network,
461 capsule network, and multi-head attention is proved. Future work will further optimize the
462 model for other downstream tasks of text classification.

References

464 Joachims, T. 1998. Text categorization with Support Vector Machines: Learning with many
465 relevant features. In European Conference on Machine Learning, 137–142.

466 Mccallum, A.; and Nigam, K. 1998. A comparison of event models for naive bayes text
467 classification. In AAAI-98 workshop on learning for text categorization, 137–142.
468 Lin, C-Y.; and Hovy, E. 2003. Automatic Evaluation of Summaries Using N-gram Co-
469 occurrence Statistics. In Proceedings of the 2003 Conference of the North American Chapter of
470 the Association for Computational Linguistics on Human Language Technology, 291–304.
471 Zhang, W.; Yoshida, T.; and Tang, X. 2008. TFIDF, LSI and multi-word in information retrieval
472 and text categorization. In IEEE International Conference on Systems, 108–113.
473 Genkin, A.; and Madigan, L. D. 2007. Large-Scale Bayesian Logistic Regression for Text
474 Categorization. 49(3): 1019–1026.
475 Pang, B.; Lee, L.; and Vaithyanathan, S. 2002. Thumbs up? Sentiment Classification using
476 Machine Learning Techniques. In Con Empirical Methods in Natural Language Processing, 79–
477 86.
478 Mikolov, T.; and Sutskever, K. e. a., I. Chen. 2013. Distributed Representations of Words and
479 Phrases and their Compositionality. 26: 3111–3119.
480 Pennington, J.; Socher, R.; and Manning, C. D. 2014. Glove: Global Vectors for Word
481 Representation. In Conferenceonference on Empirical Methods in Natural Language Processing,
482 1532–1543.
483 Kim, Y. 2014. Convolutional neural networks for sentence classification. arXiv:1408.5882.
484 Zhang, X.; Zhao, J.; and Lecun, Y. 2015. Character-level Convolutional Networks for Text
485 Classification. In Neural Information Processing Systems.
486 Conneau, A.; Schwenk, H.; Barrault, L.; and et al. 2017. Very Deep Convolutional Networks for
487 Text Classification. In European Chapter of the Association for Computational Linguistics,
488 1107–1116.
489 Mousa, A.; and Schuller, B. 2017. Contextual Bidirectional Long Short-Term Memory Recurrent
490 Neural Network Language Models: A Generative Approach to Sentiment Analysis. In European
491 Chapter of the Association for Computational Linguistics, 1023–1032.
492 Zhao, W.; Ye, J.; Yang, M.; and et al. 2018. Investigating Capsule Networks with Dynamic
493 Routing for Text Classification. In Conference on Empirical Methods in Natural Language
494 Processing, 3110–3119.
495 Sabour, S.; Frosst, N.; and Hinton, G. E. 2017. Dynamic Routing Between Capsules.
496 Xi, E.; Bing, S.; and Jin, Y. 2017. Capsule Network Performance on Complex Data.
497 arXiv:1712.03480.
498 Vaswani, A.; Shazeer, N.; and Parmar, N. e. a. 2017. Attention is All you Need.
499 Alaparthi, S.; Mishra, M. 2020. Bidirectional Encoder Representations from Transformers
500 (BERT): A sentiment analysis odyssey. arXiv: 2007.01127
501 Marcheggiani, D.; and Titov, I. 2017. Encoding sentences with graph convolutional networks for
502 semantic role labeling. In The 2017 Conference on Empirical Methods in Natural Language
503 Processing, 1506–1515.

504   Duvenaud, D. K.; and Maclaurin, J. e. a., D. Iparraguirre. 2015. Convolutional networks on
505   graphs for learning molecular fingerprints. In Neural Information Processing Systems, 2224–
506   2232.
507   Bastings, J.; Titov, I.; Aziz, W.; and et al. 2017. Graph Convolutional Encoders for Syntax-aware
508   Neural Machine Translation. In Proceedings of the 2017 Conference on Empirical Methods in
509   Natural Language Processing.
510   Chorowski, J.; Bahdanau, D.; Cho, K.; and et al. 2014. End-to-end Continuous Speech
511   Recognition using Attention-based Recurrent NN: First Results. arXiv:1412.1602.
512   Lin, Z.; Feng, M.; Santos, C. N. D.; and et al. 2017. A Structured Self-attentive Sentence
513   Embedding. In International Conference on Learning Representations.
514   Kim, Y.; Lee, H.; and Jung, K. 2018. AttnConvnet at SemEval-2018 Task 1: Attention-based
515   Convolutional Neural Networks for Multi-label Emotion Classification. In International
516   Workshop on Semantic Evaluation.
517   Eriguchi, A.; Tsuruoka, Y.; and Cho, K. 2017. Learning to parse and translate improves neural
518   machine translation. In the 55th Annual Meeting of the Association for Computational
519   Linguistics, 72–78.
520   Le, P.; and Zuidema, W. 2014. The Inside-Outside Recursive Neural Network model for
521   Dependency Parsing. In the 2014 Conference on Empirical Methods in Natural Language
522   Processing, 729–739.
523   Eriguchi, A.; Tsuruoka, Y.; and Cho, K. 2017. Learning to parse and translate improves neural
524   machine translation. In the 55th Annual Meeting of the Association for Computational
525   Linguistics, 72–78.
526   Miwa, M.; and Bansal, M. 2016. End-to-end relation extraction using lstms on sequences and
527   tree structures. In the 54th Annual Meeting of the Association for Computational Linguistics,
528   1105–1116.
529   Pang, B.; and Lee, L. 2004. A sentimental education: Sentiment analysis using subjectivity. In
530   the 54th Annual Meeting of the Association for Computational Linguistics, 271–278.
531   Hu, M.; and Liu, B. 2004. Mining and summarizing customer reviews. In Tenth Acm Sigkdd
532   International Conference on Knowledge Discovery and Data Mining.
533   Lewis, D. D. 1992. An Evaluation of Phrasal and Clustered Representations on a Text
534   Categorization Task. In Proceedings of the 15th Annual International ACM SIGIR Conference
535   on Research and Development in Information Retrieva.

Figure 1

The example of Syntactic dependency tree, where 'monkey' is the subject of the predicate 'eats', and 'apple' is its object.

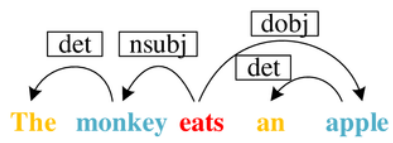



Figure 2

The architecture of the Syntax-AT-CapsNet model,which consists of three modules, Attention module,Syntax module and Capsule network module.

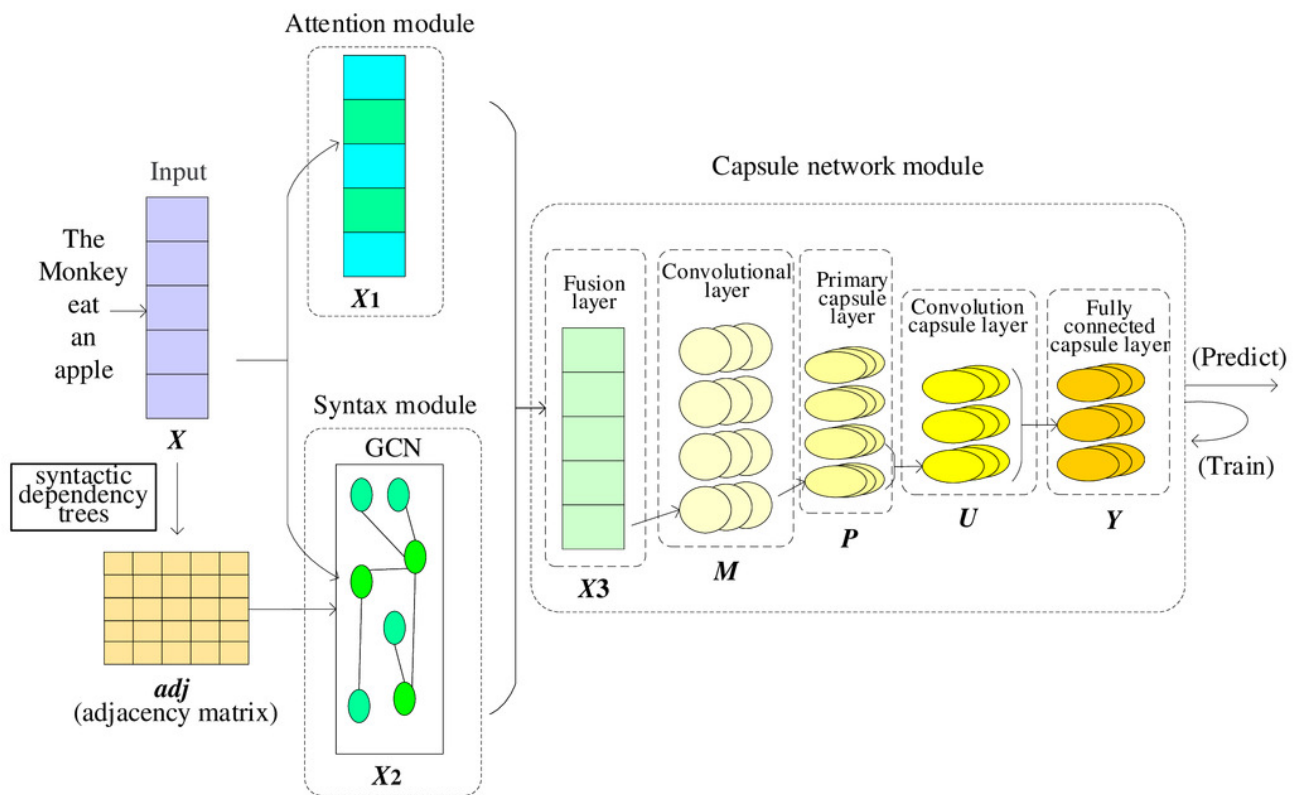



Figure 3

The architecture of the Attention module, which consists of three parallel attention computing units.

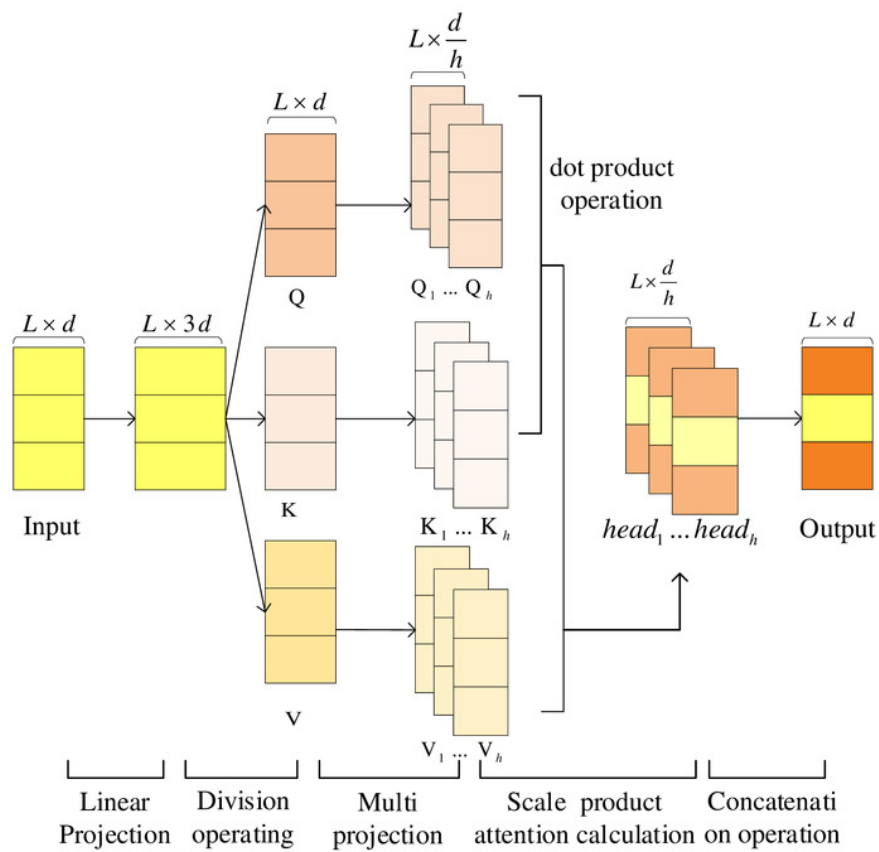



Figure 4

The architecture of the Syntax module, which includes the construction of the syntactic dependency tree and the GCN unit.

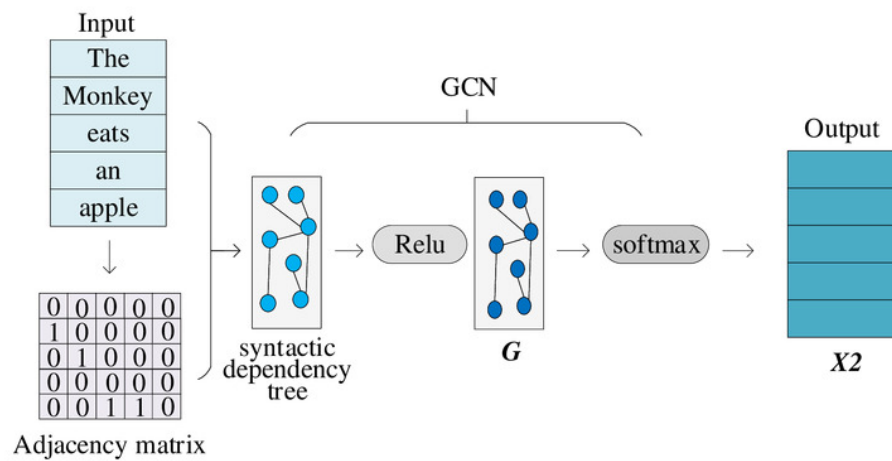



Figure 5

The example of adjacency matrix for a syntactic dependent tree.

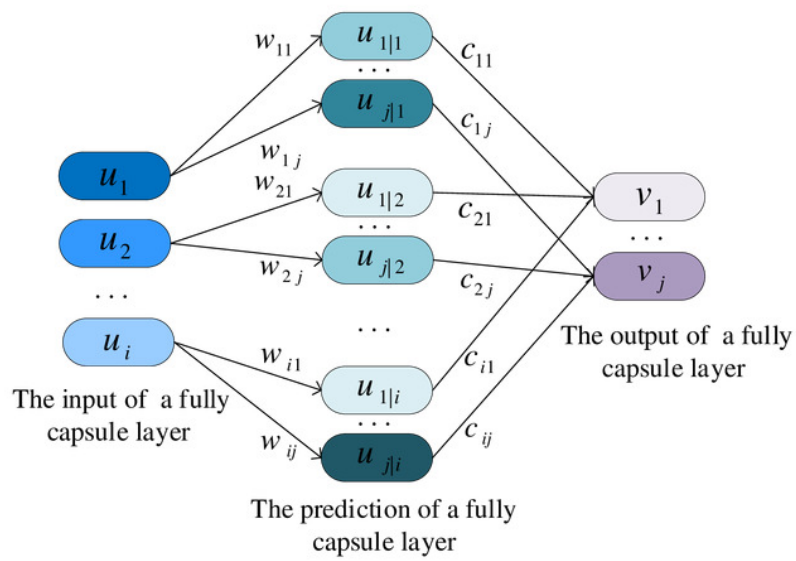
	The	Monkey	eat	an	apple
The	0	0	0	0	0
Monkey	1	0	0	0	0
eat	0	1	0	0	0
an	0	0	0	0	0
apple	0	0	1	1	0

Figure 6

The architecture of fully connected capsule layer, its input is the output of the previous capsule layer

The input of a fully
capsule layer

The prediction of a fully
capsule layer

The output of a fully
capsule layer

Data description, it describes the number of training sets, validation sets and test sets of five kinds of data sets, the number of categories of each data set, and whether the data set belongs to single-label data or multi-label data.

Data set	Train / Vec / Test	Num of category	Test label
MR	8632 / 960 / 1066	2	Single
CR	3053 / 340 / 377	2	Single
Subj	8099 / 900 / 1000	2	Multi
Reuters-Full	5000 / 500 / 2500	8	Single / Multi
Reuters-Multi	5000 / 500 / 300	8	Multi

The experimental result of single label classification. Our model is compared with the benchmark model on three single-label datasets. The evaluation index is accuracy.

Evaluation index	Accuracy(%)	Precision(%)	Recall(%)	F1(%)
Data set		MR		
TextCNN	73.0	**92.3**	87.6	89.9
Capsule-A	74.0	90.2	89.8	90.0
AT-CapsNet	74.8	91.5	90.0	90.7
Syntax-AT-Capsule(ours)	**75.3**	92.0	**90.8**	**91.4**
Data set		Subj		
TextCNN	81.6	**96.2**	88.5	92.2
Capsule-A	79.5	92.6	90.2	91.4
AT-CapsNet	82.4	92.2	91.0	91.6
Syntax-AT-Capsule(ours)	**82.9**	93.9	**92.6**	**93.2**
Data set		CR		
TextCNN	89.5	**98.2**	90.6	94.2
Capsule-A	88.7	94.0	92.0	93.0
AT-CapsNet	89.9	96.8	93.2	95.0
Syntax-AT-Capsule(ours)	**90.6**	97.3	**95.3**	**96.3**

The experimental result of multi label classification. Our model is compared with the benchmark model on two multi-label datasets. The evaluation index is ER, Precision, Recall and F1.

Evaluation index	ER(%)	Precision(%)	Recall(%)	F1(%)
Data set		Reuters-Full		
TextCNN	85.0	**97.0**	86.5	91.4
Capsule-A	84.5	92.5	90.2	91.9
AT-CapsNet	86.6	94.9	90.6	92.7
Syntax-AT-CapsNet(ours)	**88.1**	95.0	**91.8**	**93.4**
Data set		Reuters-Multi		
TextCNN	28.7	**96.0**	59.1	73.2
Capsule-A	44.3	86.3	75.6	80.6
AT-CapsNet	64.3	88.5	84.6	86.5
Syntax-AT-CapsNet(ours)	**70.3**	95.2	**84.9**	**89.7**

The experimental result of Syntax module verification. Our syntactic module was added to the benchmark model and experimented on Reuters-Full datasets. The evaluation index is ER, Precision, Recall and F1. In each control group, the first model has no syn

Data set	Reuters-Full			
Evaluation index	ER(%)	Precision(%)	Recall(%)	F1(%)
TextCNN	85.0	97.0	86.5	91.4
Syntax-CNN	**85.9**	**97.1**	**87.7**	**92.1**
Capsule-A	84.5	92.5	90.2	91.9
Syntax-CapsNet	**85.4**	**93.1**	**91.7**	**92.4**
AT-CapsNet	86.6	94.9	90.6	92.7
Syntax-AT-CapsNet(ours)	**88.1**	**95.0**	**91.8**	**93.4**

The experimental result of control single module ablation. Remove one module from our model to verify the effect of the module. There is no corresponding module for the first model in each control group.

Data set	Reuters-Full			
Evaluation index	ER(%)	Precision(%)	Recall(%)	F1(%)
With/without syntax module				
AT-CapsNet	86.6	94.9	90.6	92.7
Syntax-AT-CapsNet(ours)	**88.1**	**95.0**	**91.8**	**93.4**
Decrease value	1.5	0.1	1.2	0.7
With/without attention module				
Syntax-CapsNet	85.4	93.1	91.7	92.4
Syntax-AT-CapsNet(ours)	**88.1**	**95.0**	**91.8**	**93.4**
Decrease value	2.7	1.9	0.1	1
With/without capsule network module				
Syntax-AT-CNN	85.2	94.0	91.0	92.5
Syntax-AT-CapsNet(ours)	**88.1**	**95.0**	**91.8**	**93.4**
Decrease value	2.9	1	0.8	0.9

The experimental result of control two module ablation. Remove two module from our model to verify the effect of the module. There is no corresponding module for the first model in each control group.

Data set	Reuters-Full			
Evaluation index	ER(%)	Precision(%)	Recall(%)	F1(%)
With/without syntax + attention module				
CapsNet	84.5	92.5	90.2	91.9
Syntax-AT-CapsNet(ours)	88.1	95.0	91.8	93.4
Decrease value	3.6	2.5	1.6	1.5
With/without syntax + capsule network module				
AT-CNN	84.0	94.3	87.5	90.8
Syntax-AT-CapsNet(ours)	88.1	95.0	91.8	93.4
Decrease value	4.1	0.7	4.3	2.6
With/Without Attention + Capsule Network Module				
Syntax-CNN	85.9	97.1	87.7	92.1
Syntax-AT-CapsNet(ours)	88.1	95.0	91.8	93.4
Decrease value	2.2	-2.1	4.1	1.3