Dynamically Route Hierarchical Structure Representation to Attentive Capsule for Text Classification

Wanshan Zheng[1,2] , **Zibin Zheng**[1,2] , **Hai Wan**[1] , **Chuan Chen**[1,2]
[1]School of Data and Computer Science, Sun Yat-sen University, Guangzhou, China
[2]Guangdong Key Laboratory for Big Data Analysis and Simulation of Public Opinion, The School of Communication and Design, Sun Yat-sen University, Guangzhou, China
zhengwsh3@mail2.sysu.edu.cn, {zhzibin, wanhai, chenchuan}@mail.sysu.edu.cn

Abstract

Representation learning and feature aggregation are usually the two key intermediate steps in natural language processing. Despite deep neural networks have shown strong performance in the text classification task, they are unable to learn adaptive structure features automatically and lack of a method for fully utilizing the extracted features. In this paper, we propose a novel architecture that dynamically routes hierarchical structure feature to attentive capsule, named HAC. Specifically, we first adopt intermediate information of a well-designed deep dilated CNN to form hierarchical structure features. Different levels of structure representations are corresponding to various linguistic units such as word, phrase and clause, respectively. Furthermore, we design a capsule module using dynamic routing and equip it with an attention mechanism. The attentive capsule implements an effective aggregation strategy for feature clustering and selection. Extensive results on eleven benchmark datasets demonstrate that the proposed model obtains competitive performance against several state-of-the-art baselines. Our code is available at https://github.com/zhengwsh/HAC.

1 Introduction

Text classification is one of the fundamental tasks in natural language processing (NLP). In recent years, many successful deep learning models have been widely applied to this task. Mainstream deep learning methods usually contain two important components: *representation learning* and *feature aggregation*. Given a sentence or document in the classification task, the longstanding challenges for deep learning models are (1) how to enumerate task-relevant structure representations of the sequence automatically via representation learning and (2) how to leverage latent label-oriented abstractions of each extracted structure feature via feature aggregation.

According to different representation learning approaches, existing models for text classification can be roughly categorized into four types. *Bag-of-words representation models* represent the text sequence by taking the average of different words [Grave *et al.*, 2017], but unable to consider word order. *Sequence representation models* consider word order using convolutional neural network [Kim, 2014] or recurrent neural network [Chung *et al.*, 2014], but do not involve structure information. *Structure representation models*, such as tree-structured LSTM [Tai *et al.*, 2015], utilize pre-specified parsing trees to take structure information into account. *Attention-based models* [Yang *et al.*, 2016] use attention mechanism to build representations by weighting input words differently.

Currently, the structure features in most structure representation models are either provided as input from pre-defined parsers or learned end-to-end from feature detectors, such as CNN and RNN. However, these representation methods are not automatical and unable to learn adaptive text structure sufficiently. Focusing on CNN encoder, the traditional pipeline is using convolutional layers with geometrically fixed filters to extract spatial features (n-gram structure) in the last level of output, whatever for shallow or deep encoders. Then a following pooling or attention layer is utilized to aggregate prominent features for downstream tasks. The main limitations include that little structure information is explicitly learned from the encoder, and the aggregation strategies perform selection directly over structure representations, which can't fully utilize their latent semantic abstractions.

More recently, a promising work named capsule network [Sabour *et al.*, 2017] provides novel viewpoint of feature aggregation. The dynamic routing process learns part-whole relation between adjacent capsule layers. However, existing capsule networks determine the category either according to the vector length of capsules or just flatten all capsules to a fixed-length vector for the classifier, which lost potential and diverse information embedded in different capsules.

In this paper, to address the aforementioned challenges and issues, we propose a novel model (dubbed as **HAC**) that dynamically route **H**ierarchical structure feature to **A**ttentive **C**apsule for text classification. We employ a deep dilated CNN to extract hierarchical structure features. Different from conventional CNNs, we denote the intermediate feature maps at each level of deep dilated CNN as hierarchical structure representations. Dilated convolutions exponentially grow the receptive field from upstream layers to downstream layers, which can effectively form various linguistic units of input text, such as word, phrase, clause, sentence and other specific levels. To aggregate and fully utilize information within hierarchical structure features, we design a capsule module using

dynamic routing and equip it with an attention mechanism. During the dynamic routing process, information is embedded into capsules and iteratively distilled into task-relevant categories, which can be viewed as information distillation and feature clustering. Considering that target capsules contain different viewpoints and contribute unequally to the text category, we build an attention mechanism upon the aggregation of target capsules to adaptively select significant parts for classification. Extensive experiments on several benchmarks demonstrate the effectiveness of the proposed architecture, surpassing the state-of-the-art methods remarkably.

Specifically, our contributions are of three-folds:

- We propose a novel HAC architecture for improving the structure representation learning and feature aggregation within the text classification task, by dynamically routing hierarchical structure feature to attentive capsule.
- We design an effective CNN-based module to automatically extract hierarchical representations of text structure, and a variant capsule-based module equipped with an attention mechanism to make full use of information.
- We conduct extensive experiments on both small and large datasets, covering a wide range of text classification tasks. Experiments demonstrate that our proposed approach outperforms a number of competitive baselines and achieve a few state-of-the-art results.

2 Related Work

Related work can be divided into two threads and we briefly review each of these areas in this section.

2.1 Structure Representation

Structure representation is an attractive representation method for sentence modeling. In the classification task, existing works can be concluded into three types. Hierarchical models include [Zhou *et al.*, 2015], combining CNN and RNN to learn structure representation. Tree-based models include [Tai *et al.*, 2015], involving pre-defined parsers to construct sentence tree structure. Reinforcement learning models include [Tianyang *et al.*, 2018], using policy network and classification network to discover optimized structures.

However, previous models only select the final level output of encoder (CNN/RNN) as sentence representations and pass it to the following classifier. To utilize multi-scale structure features within the sentence, [Zhao *et al.*, 2015; Wang *et al.*, 2018b] use intermediate information obtained during the encoding process. [Zhao *et al.*, 2015] concatenates different representations of the sentences at different levels of abstractions, by performing a pooling operation over each level of recurrent and recursive networks. [Wang *et al.*, 2018b] introduces a densely connected CNN and an attention mechanism to choose each level of CNN features.

2.2 Feature Aggregation

Feature aggregation aims at aggregating the extracted features into a fixed-length vector as sentence representation. Bag-of-word (BoW) [Grave *et al.*, 2017] strategy represents the sentence by taking the average of embedding vectors. Max-pooling is employed to extract the most salient feature within each feature map in CNNs [Kim, 2014; Zheng *et al.*, 2017] or within each hidden state in RNNs. Dynamic k-max pooling [Kalchbrenner *et al.*, 2014] strategy selects the k most active features to prevent missing valuable information. Attention mechanism becomes popular by calculating attention distribution and performing a weighted average on the output features of CNN/RNN layer [Yang *et al.*, 2016], for classification [Chen *et al.*, 2018] and transfer learning [Shaoan *et al.*, 2018].

Recently, capsule network with dynamic routing is proposed to improve the representational limitations of CNNs and RNNs. Capsule network is first applied in image [Sabour *et al.*, 2017], and recently in text [Zhao *et al.*, 2018; Gong *et al.*, 2018]. Unlike pooling operation, the capsule network collects the whole information after transformation, instead of discarding unrelated information.

3 Method

The overall architecture of our model is depicted in Figure 1. Following sections elaborate the components and workflow.

3.1 Word Embedding Layer

Consider a text sentence $S = w_1, ..., w_N$ with length N. The goal of this layer is to represent each word in S with a d-dimensional vector. The word embedding is a fixed vector for each individual word, which is projected from the pre-trained embedding lookup-table. The output of this layer is the sequence of word vectors $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_N] \in \mathbf{R}^{N \times d}$.

3.2 Context Representation Layer

The goal of this layer is to incorporate contextual information into the representation of each input time step. Meanwhile, the word vectors will be compressed into a lower dimension for the following layers. We utilize a bi-directional Gated Recurrent Unit (Bi-GRU) [Cho *et al.*, 2014] to produce contextual representation $\mathbf{h}_t$ of a word by concatenating forward hidden state output $\overrightarrow{\mathbf{h}_t}$ and backward hidden state output $\overleftarrow{\mathbf{h}_t}$.

$$\overrightarrow{\mathbf{h}_t} = \overrightarrow{GRU}\ (\overrightarrow{\mathbf{h}_{t-1}}, \mathbf{x}_t), \tag{1}$$

$$\overleftarrow{\mathbf{h}_t} = \overleftarrow{GRU}\ (\overleftarrow{\mathbf{h}_{t+1}}, \mathbf{x}_t), \tag{2}$$

$$\mathbf{h}_t = [\overrightarrow{\mathbf{h}_t}, \overleftarrow{\mathbf{h}_t}]. \tag{3}$$

Thus, the output of Bi-GRU encoder are a sequence of vectors $\mathbf{H} = [\mathbf{h}_1, ..., \mathbf{h}_N] \in \mathbf{R}^{N \times d'}$.

3.3 Deep Dilated Convolution Layer

This is one of the core layers within our model. The purpose of this deep dilated convolution layer is to extract hierarchical multi-granularity features as textual structure representation.

Unlike conventional CNN which apply convolution operation upon pre-trained word embeddings, our dilated CNN is built upon Bi-GRU output vectors which contain contextual information. For the first convolution block, we denote $\mathbf{U}^0 = \mathbf{H} = [\mathbf{h}_1, ..., \mathbf{h}_N] \in \mathbf{R}^{N \times d'}$ as the input representation matrix, where d' is the input vector dimension. Similarly,

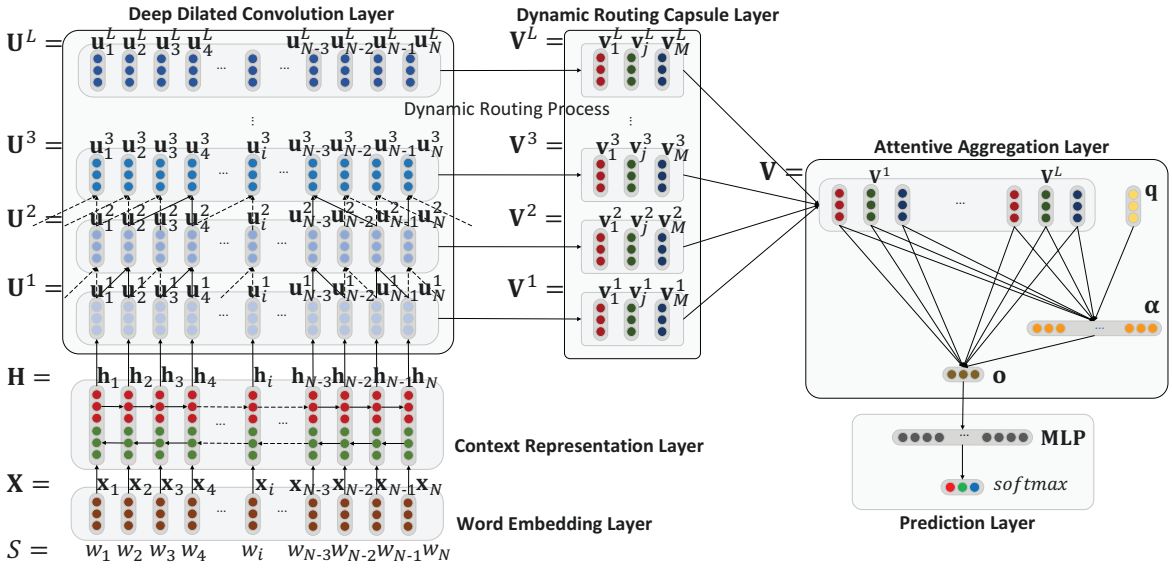

Figure 1: (Better viewed in color) The overview of our proposed architecture HAC.

the outputs of each intermediate block and the final block can be represented as $\mathbf{U}^l = [\mathbf{u}_1^l, ..., \mathbf{u}_N^l] \in \mathbf{R}^{N\times k} (l \in [1, L])$, where L is the number of total convolution block and k is the number of filters in each block.

Consider the l-th convolution block, let $\mathbf{W}^l \in \mathbf{R}^{k\times w\times k}$ ($\mathbf{W}^1 \in \mathbf{R}^{k\times 1\times d'}$ is the exception) be the filter matrix for convolving w input vectors with k filter kernels. The transformation between two adjacent blocks can be formulated as

$$\mathbf{U}^l = f(\mathbf{W}^l, \mathbf{U}^{l-1}), \tag{4}$$

where f is an affine function that sliding filters over the w-length input window.

Specifically, $\mathbf{u}_t^l \in \mathbf{U}^l$ is produced by calculating

$$\mathbf{u}_t^l = ReLU(\mathbf{W}^l \bigoplus [\mathbf{u}_{t+ir}^{l-1}]_{i=0}^{w-1}), \tag{5}$$

where [] is vector concatenation operator, $\bigoplus$ is convolution operator and r is dilated rate. Rectified linear units ($ReLU$) function is adopted for activation.

We use a dilation scheme whereby the dilation rates are doubled every block up to a maximum rate 2^{L-2} and thus the receptive fields are increased every block up to maximum width $(w-1)2^{L-1}$, which can be fine-tuned according to specific dataset. Notice that in the standard convolution, $r = 1$.

Dilation [Yu and Koltun, 2016] makes the receptive field grow exponentially in terms of the depth of the networks, as opposed to linearly. The receptive field can be computed by $(w-1)2^{l-1}$. Using larger dilation rate or larger filter size enable the output at each intermediate level to represent a wider range of input, which can be regarded as a larger n-gram feature. Intuitively, variant size of n-gram features corresponding to multi-granularity structure representation of the input text sequence, where small n-gram match word/phrase-level and large n-gram match clause/sentence-level in general.

Zero padding is used on either side of the convolution layer's input to ensure that the output features have the same size as the input features. Finally, we extract hierarchical feature maps $\mathbf{U}^1, \mathbf{U}^2, ..., \mathbf{U}^L$.

3.4 Dynamic Routing Capsule Layer

This is another core layer within our model. The goal of this layer is to summarize the raw convolutional features generated at the previous layer into several higher abstract aspects. We propose a dynamic routing policy, a variant of capsule network first proposed by [Sabour *et al.*, 2017], to implement this process for feature clustering. In this layer, upstream row features and downstream abstract features are both represented in capsules. Note that for each convolution block, an unshared capsule layer is adopted to categorize their output features, with regard to a specific linguistic unit.

As shown in Figure 2, we take one convolution block's output features for example and formally introduce the dynamic routing process in detail. Recall that $\mathbf{U}^l = [\mathbf{u}_1^l, ..., \mathbf{u}_i^l, ..., \mathbf{u}_N^l] \in \mathbf{R}^{N\times k}$ denotes the output of l-th convolution block. Each $\mathbf{u}^l$ represents a n-gram feature generated by k filter maps and we call it as source capsule. And we call $\mathbf{V}^l = [\mathbf{v}_1^l, ..., \mathbf{v}_j^l, ..., \mathbf{v}_M^l] \in \mathbf{R}^{M\times d_v}$ as target capsules, where M denotes the target capsule number and d_v denotes the target capsule dimension. We aim at routing $\mathbf{U}^l$ to $\mathbf{V}^l$, as an information distillation and feature clustering process. For simplicity of notation, we omit the superscript l in the following formulas.

Consider $\mathbf{u}_i$ in source capsule layer and $\mathbf{v}_j$ in target capsule layer, a routing vector $\mathbf{m}_{i\to j}$ is computed to indicate the information to be transferred from $\mathbf{u}_i$ to $\mathbf{v}_j$

$$\mathbf{m}_{i\to j} = c_{ij}\hat{\mathbf{u}}_{j|i}. \tag{6}$$

Prediction vector (vote) $\hat{\mathbf{u}}_{j|i}$ indicates the viewpoint of raw feature to be transferred and is computed from $\mathbf{u}_i$ by multiplying a transformation matrix $\mathbf{W}_j$

$$\hat{\mathbf{u}}_{j|i} = \mathbf{W}_j\mathbf{u}_i. \tag{7}$$

Coupling coefficients c_{ij} indicates the proportion of prediction vector to be transferred and is determined by a

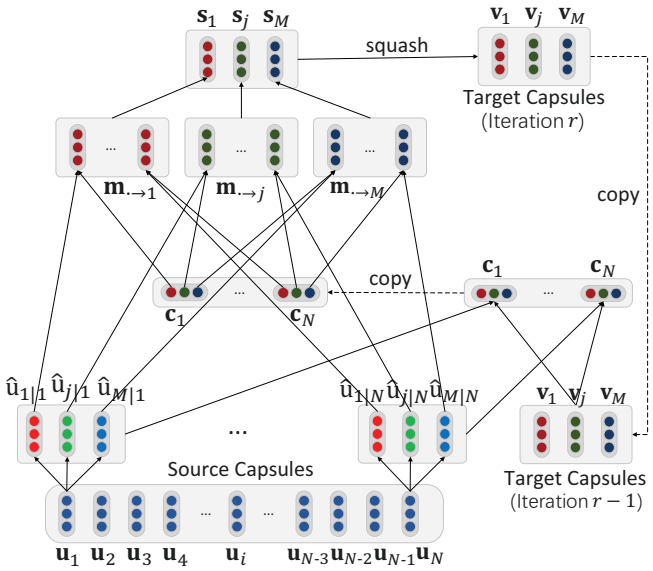

Figure 2: (Better viewed in color) Dynamic routing process

"routing softmax" function

$$c_{ij} = \frac{exp(b_{ij})}{\sum_k exp(b_{ik})}, \tag{8}$$

where b_{ij} represents the log prior probability that capsule i should be coupled to capsule j and is calculated by the iterative dynamic routing process.

Then, the target capsule $\mathbf{v}_j$ is produced with two steps. A collection of incoming routing vectors is calculated firstly

$$\mathbf{s}_j = \sum_{i=1}^{N} \mathbf{m}_{i \to j}, \tag{9}$$

and then be activated by applying the squash function

$$\mathbf{v}_j = \frac{\mathbf{s}_j}{\|\mathbf{s}_j\|}. \tag{10}$$

The squash function enlarges small vectors and shrinks large vectors to unit vectors, which makes the dynamic routing process more stable and the information interaction more efficient.

Between two capsule layers, we use an iterative dynamic routing policy to compute the capsule output by calculating an intermediate value c_{ij}. Within the "routing softmax" function, b_{ij} is initialized with zero and updated with an agreement scale a_{ij}. The agreement a_{ij} is calculated by a scale product between capsules in two layers

$$a_{ij} = \mathbf{v}_j \hat{\mathbf{u}}_{j|i}, \tag{11}$$

$$b_{ij} \leftarrow b_{ij} + a_{ij}. \tag{12}$$

By default, the target capsule number M is the number of text categories plus an orphan category. The orphan category inspired by [Sabour *et al.*, 2017] can capture the "background" information such as stop words and the words that are unrelated to the task. Introducing the orphan category reduces the interference for normal categories and makes the dynamic routing more effective.

Algorithm 1 Dynamic Routing Algorithm

Input: Source capsules $\mathbf{u}_1, ..., \mathbf{u}_N$; Maximum iterations R
Output: Target capsules $\mathbf{v}_1, ..., \mathbf{v}_M$

```
Initialize b_ij = 0 for all i and j.
for r = 1 to R do
    Compute c_ij for all i and j by Eq. (8)
    Update all target capsules v_j by Eq. (10)
    Update b_ij for all i and j by Eq. (12)
end for
return v_1, ..., v_j, ..., v_M
```

Dataset	c	l	ml	N	Test	Type
MR	2	22	63	10662	CV	review
SST-1	5	19	56	11855	2210	sentiment
SST-2	2	19	56	9613	1821	sentiment
SUBJ	2	25	132	10000	CV	subjectivity
TREC	6	10	37	5952	500	question
CR	2	20	106	3775	CV	review
MPQA	2	3	44	10606	CV	opinion
AG	4	36	199	127.6k	7.6k	news
DBP	14	52	1498	630k	70k	ontology
Yelp.P	2	153	1221	598k	38k	review
Yelp.F	5	154	1221	700k	50k	review

Table 1: Statistics of eleven datasets after tokenization (top half are **small** datasets and bottom half are **large** datasets). c: number of classes. l: average sentence length. ml: maximum sentence length. N: dataset size. Test: test set size (CV means no standard train/test split and thus nested 10-fold cross validation is used).

Our dynamic routing algorithm is summarized in Algorithm 1. Each separated target capsule layer yields $\mathbf{V}^l = [\mathbf{v}_1^l, ..., \mathbf{v}_j^l, ..., \mathbf{v}_M^l]$ for the l-th convolutional feature block. We concatenate target capsules outputs into a united

$$\mathbf{V} = [\mathbf{V}^1, \mathbf{V}^2, ...]. \tag{13}$$

and pass it to the attentive aggregation layer.

3.5 Attentive Aggregation Layer

The goal of this layer is producing a fixed-length representation vector $\mathbf{o}$ by taking all target capsules as input. For each target capsule $\mathbf{v}_i \in \mathbf{R}^{d_v}$ in $\mathbf{V}$, we compute an attention score α_i for it, which indicates its importance and contribution to the classification task,

$$\alpha_i = \frac{exp(e_i)}{\sum_k exp(e_k)}, \tag{14}$$

$$e_i = a(\mathbf{q}, \mathbf{v}_i), \tag{15}$$

$$a(q, v_i) = \mathbf{q}^{\mathbf{T}} \mathbf{v}_i, \tag{16}$$

where $\mathbf{q}$ is a task-specific trainable pattern vector and k denotes the number of capsules in the capsule pool $\mathbf{V}$.

Finally, we obtain a fix-length aggregation vector $\mathbf{o}$ by calculating the weighted sum over all target capsules for text representation and pass it to the downstream classifier.

$$\mathbf{o} = \sum\nolimits_i \alpha_i \mathbf{v}_i \tag{17}$$

	Model	MR	SST-1	SST-2	Subj	TREC	CR	MPQA
RNNs (no transfer)	Bi-LSTM	79.3	46.2	83.2	90.5	89.6	82.1	-
	Tree-LSTM	80.7	**50.1**	85.7	91.3	91.8	83.2	-
CNNs (no transfer)	MC-CNN	81.1	47.4	88.1	93.2	92.2	85.0	89.4
	MVCNN	-	49.6	**89.4**	93.9	-	-	-
Others (no transfer)	CapsuleB	82.3	-	86.8	93.8	92.8	85.1	-
	Self-ATT	80.1	47.2	-	92.5	-	-	-
Ours (no transfer)	HAC	**83.3**	49.1	88.2	**95.1**	**95.0**	**86.4**	**89.8**
Transfer Approaches	BoW+ELMo	79.7	48.7	86.3	94.3	93.4	85.1	89.6
	InferSent	81.1	-	84.6	92.4	88.2	86.3	90.2
	USE_T+CNN	81.2	-	86.7	93.6	**98.1**	87.5	87.3
Ours	HAC+ELMo	**85.0**	**49.7**	**89.4**	**95.9**	96.8	**88.9**	**91.2**

Table 2: Experimental result (test set accuracy) comparison of our model and baselines on seven **small** text classification benchmarks.

3.6 Prediction Layer

The purpose of this layer is to estimate the probability distribution $\mathbf{p}(\mathbf{y}|S)$, where $\mathbf{y}$ is the classification targets. We feed the fixed-length aggregation vector $\mathbf{o}$ to a multi-layer perceptron (MLP) classifier using *softmax* activation.

$$\mathbf{p}(\mathbf{y}|S) = \mathbf{p}(\mathbf{y}|\mathbf{o}) = softmax(\mathbf{MLP}(\mathbf{o})) \quad (18)$$

4 Experiments

4.1 Datasets

Literature usually tests their model on either small datasets or large datasets, which could not provide extensive results. Here, we test our model on both. For **small** datasets, seven widely-studied datasets [Kim, 2014] include: movie reviews (MR), Stanford Sentiment Treebank (SST-1 and SST-2), subjectivity classification (SUBJ), question dataset (TREC), customer review (CR), opinion polarity (MPQA). For **large** datasets, four widely-studied datasets [Zhang *et al.*, 2015] include: AG's news corpus (AG), DBPedia ontology (DBP), Yelp reviews (Yelp.P and Yelp.F).

The detailed statistics are listed in Table 1. These datasets cover a wide range of text classification tasks, with varying numbers of documents and varying document length. Following the evaluation scheme in existing literatures, for MR, CR, Subj, MPQA, we use nested 10-fold cross-validation, for TREC, AG, DBP, Yelp.P, Yelp.F, 10-fold cross-validation, and for SST-1, SST-2, standard validation.

4.2 Implementation Details

Input Embedding

We use two versions of embedding for fair comparisons. The first one is 300-dimensional GloVe [Pennington *et al.*, 2014] word vectors trained on Common Crawl corpus. Another is 1024-dimensional word vectors extracted from ELMo [Peters *et al.*, 2018] pre-trained on Word Benchmark corpus. For both small and large datasets, GloVe embedding is adopted. Besides, ELMo embedding is adopted on small datasets for fairly comparing with results in existing transfer learning literatures. Word embeddings are fixed during training.

Model Configuration

We are interested in our model's robustness across a diverse set of tasks. To this end, if not mentioned otherwise, we use the same network parameters in all classification tasks without specific fine-tuning. In detail, we set the hidden state dimension of Bi-GRU to be 100 for each direction. We adopt 5 dilated convolutional blocks, with filter window size 2 and filter number 100. We set the target capsule dimension to be 50, and use 3 iterations of routing for all datasets. The MLP classifier has a hidden layer of size 50 using ReLU activation.

Training Protocol

The training objective is to minimize the cross-entropy loss. Dropout regularization is employed on the input embedding layer, with the dropout rate 0.5. We don't impose L2 regularization at each layer. We train our model's parameters using gradient-based optimizer Adam, with an initial learning rate 1e-3. We halve the learning rate if the dev accuracy doesn't increase in 3 training epochs, and set the minimum rate to be 1e-4. We conduct mini-batch with size 8 for small datasets, and size 128 for large datasets. The training process lasts at most 30 epochs on all the datasets.

4.3 Competitors

To comprehensively evaluate the performance of our proposed approach, we give a variety of baseline methods and state-of-the-art models for comparison, listed in different genres (RNNs variants, CNNs variants, and others).

For **small** datasets, the competitors include: Bi-LSTM [Cho *et al.*, 2014], tree-structured LSTM (Tree-LSTM) [Tai *et al.*, 2015], multichannel CNN (MC-CNN) [Kim, 2014], multichannel variable-size CNN (MVCNN) [Yin and Schütze, 2015], capsule network (CapsuleB) [Zhao *et al.*, 2018], structured self-attentive model (Self-ATT) [Lin *et al.*, 2017]. Additionally, transfer learning methods include bag-of-word with ELMo (BoW+ELMo) [Perone *et al.*, 2018], sentence representations from inference (InferSent) [Conneau *et al.*, 2017a], transformer-based universal sentence encoder (USE_T+CNN) [Cer *et al.*, 2018]. For **large** datasets, the

	Model	AG	DBP.	Yelp.P	Yelp.F
RNNs	LSTM	86.1	98.6	94.7	52.5
	Dis-LSTM	92.1	98.7	92.6	59.6
CNNs	VD-CNN	91.3	98.7	95.7	64.7
	DC-CNN	93.6	**99.2**	96.5	66.0
Others	LEAM	92.5	99.0	95.3	64.1
	WC-RE	92.8	98.9	96.4	64.9
Ours	HAC	**93.7**	**99.2**	**97.4**	**68.3**

Table 3: Experimental result (test set accuracy) comparison of our model and baselines on four **large** text classification benchmarks.

Models	SST-2 Acc.	AG Acc.
Full model	**88.2**	**93.7**
w/o intermediate features	86.5	92.8
w/o capsule routing	87.6	93.3
w/o attentive aggregation	87.4	93.0

Table 4: Ablation study of our model on dataset SST-2 and AG.

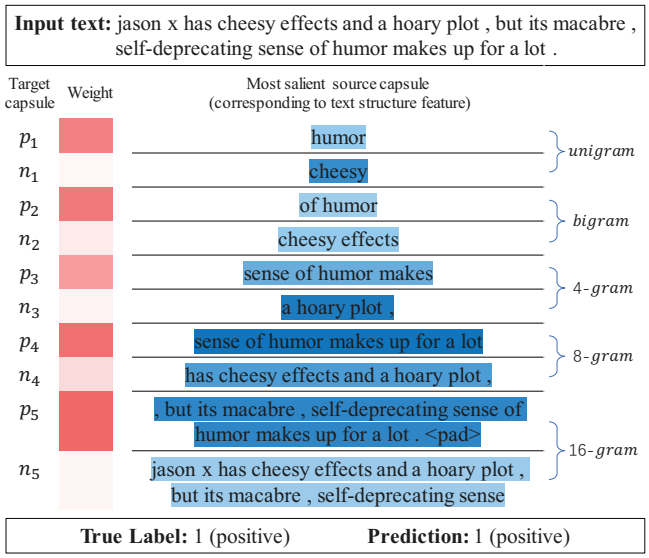

Figure 3: Visualization of structure features and attentive capsules. Type notation p is positive category, n is negative category and the subscript implies corresponding layer.

competitors include: LSTM and its variant discriminative LSTM (Dis-LSTM) [Yogatama *et al.*, 2017], very deep CNN (VD-CNN) [Conneau *et al.*, 2017b], densely connected CNN (DC-CNN) [Wang *et al.*, 2018b], label-embedding attentive model (LEAM) [Wang *et al.*, 2018a], word-context region embedding (WC-RE) [Qiao *et al.*, 2018].

4.4 Main Results

Classification results show the strong generalization capability of our model, across different datasets and different tasks.

Small Datasets Results

The results of small datasets are listed in Table 2. From the results, we observe that our model achieves best classification accuracy on 5 out of 7 benchmarks under the non-transfer scheme, and 6 of 7 benchmarks under the transfer scheme. Particularly, our model outperforms previous deep neural network models with a large margin and beats advanced models involving capsule or self-attention. Notice that our model under the non-transfer scheme has surpassed some state-of-the-art transfer learning approaches on several datasets.

Large Datasets Results

The results of large datasets are listed in Table 3. From the results, we observe that our model achieves best classification accuracy on all four benchmarks. Particularly, our model significantly outperforms RNNs and other advanced approaches on all datasets. While comparing with those deep and sophisticated CNNs, our model achieves analogical improvement, especially for datasets with larger average word length.

4.5 Ablation Study

More investigations are conducted to study the independent effect of each module in our proposed architecture. We replace intermediate features with the final-layer feature, capsule routing with max-pooling, attentive aggregation with concatenation, respectively. Table 4 shows the accuracy on the SST-2 (small dataset) test set and the AG (large dataset) test set. We can observe that eliminating any of the proposed modules would hurt the performance significantly. Ablation results reveal the effectiveness of each proposed component and the elaborate full architecture among different tasks.

4.6 Visualization

To intuitively understand our model, we visualize the extracted most salient structure feature and attention weight for each target capsule. Figure 3 shows the visualization result for a sentence sampled from SST-2 dataset. Red denotes the target capsule attention weight according to Equation (14), and blue denotes the source capsule coupling coefficient towards current target capsule according to Equation (8). We sort different levels of positive/negative categories from top to down. Due to limited space, we omit orphan categories, which always have small attention weights and unrelated structure. From this figure, we can see that the detected most salient text structure feature mostly represents a linguistic unit and carries corresponding sentiment polarity to their target capsule. In addition, the aggregation mechanism is able to select the informative target capsules containing strong sentiment corresponding to the true label for classification.

5 Conclusion

In this paper, we propose a novel architecture that dynamically route hierarchical structure feature to attentive capsule for text classification. The main idea of the proposed model is to adaptively form multi-granularity structure representations of text and fully leverage the categorized abstract of features with attention. Experiments on various datasets demonstrate that the proposed approach outperforms competitors and achieve several state-of-the-art results. Ablation and visualization analyses also reveal the effectiveness of our model for structure representation learning and feature aggregation.

Acknowledgments

This paper was supported by the National Key Research and Development Program (2016YFB1000101), the National Natural Science Foundation of China (61722214, 61573386, 11801595) and the Natural Science Foundation of Guangdong (2018A030310076, 2016A030313292). Corresponding authors are Hai Wan and Chuan Chen.

References

[Cer *et al.*, 2018] Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, et al. Universal sentence encoder. *arXiv preprint arXiv:1803.11175*, 2018.

[Chen *et al.*, 2018] Chuan Chen, Jingxue Xin, et al. A semisupervised classification approach for multidomain networks with domain selection. *IEEE TNNLS*, 2018.

[Cho *et al.*, 2014] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *EMNLP*, pages 1724–1734, 2014.

[Chung *et al.*, 2014] Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NIPS 2014 Workshop on Deep Learning*, 2014.

[Conneau *et al.*, 2017a] Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. Supervised learning of universal sentence representations from natural language inference data. In *EMNLP*, 2017.

[Conneau *et al.*, 2017b] Alexis Conneau, Holger Schwenk, Loıc Barrault, and Yann Lecun. Very deep convolutional networks for text classification. In *EACL*, 2017.

[Gong *et al.*, 2018] Jingjing Gong, Xipeng Qiu, Shaojing Wang, and Xuanjing Huang. Information aggregation via dynamic routing for sequence encoding. *arXiv preprint arXiv:1806.01501*, 2018.

[Grave *et al.*, 2017] Edouard Grave, Tomas Mikolov, Armand Joulin, and Piotr Bojanowski. Bag of tricks for efficient text classification. In *EACL*, page 427, 2017.

[Kalchbrenner *et al.*, 2014] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. A convolutional neural network for modelling sentences. In *ACL*, 2014.

[Kim, 2014] Yoon Kim. Convolutional neural networks for sentence classification. In *EMNLP*, 2014.

[Lin *et al.*, 2017] Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. A structured self-attentive sentence embedding. In *ICLR*, 2017.

[Pennington *et al.*, 2014] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *EMNLP*, 2014.

[Perone *et al.*, 2018] Christian S Perone, Roberto Silveira, and Thomas S Paula. Evaluation of sentence embeddings in downstream and linguistic probing tasks. *arXiv preprint arXiv:1806.06259*, 2018.

[Peters *et al.*, 2018] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *NAACL*, pages 2227–2237, 2018.

[Qiao *et al.*, 2018] Chao Qiao, Bo Huang, Guocheng Niu, Daren Li, Daxiang Dong, Wei He, et al. A new method of region embedding for text classification. In *ICLR*, 2018.

[Sabour *et al.*, 2017] Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. Dynamic routing between capsules. In *NIPS*, pages 3856–3866, 2017.

[Shaoan *et al.*, 2018] Xie Shaoan, Zheng Zibin, et al. Learning semantic representations for unsupervised domain adaptation. In *ICML*, pages 5419–5428, 2018.

[Tai *et al.*, 2015] Kai Sheng Tai, Richard Socher, et al. Improved semantic representations from tree-structured long short-term memory networks. In *ACL*, 2015.

[Tianyang *et al.*, 2018] Zhang Tianyang, Huang Minlie, and Zhao Li. Learning structured representation for text classificationxvia reinforcement learning. In *AAAI*, 2018.

[Wang *et al.*, 2018a] Guoyin Wang, Chunyuan Li, Wenlin Wang, Yizhe Zhang, Dinghan Shen, Xinyuan Zhang, Ricardo Henao, and Lawrence Carin. Joint embedding of words and labels for text classification. In *ACL*, 2018.

[Wang *et al.*, 2018b] Shiyao Wang, Minlie Huang, and Zhidong Deng. Densely connected cnn with multi-scale feature attention for text classification. In *IJCAI*, 2018.

[Yang *et al.*, 2016] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, et al. Hierarchical attention networks for document classification. In *NAACL*, 2016.

[Yin and Schütze, 2015] Wenpeng Yin and Hinrich Schütze. Multichannel variable-size convolution for sentence classification. In *CoNLL*, pages 204–214, 2015.

[Yogatama *et al.*, 2017] Dani Yogatama, Chris Dyer, Wang Ling, and Phil Blunsom. Generative and discriminative text classification with recurrent neural networks. *arXiv preprint arXiv:1703.01898*, 2017.

[Yu and Koltun, 2016] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. In *ICLR*, 2016.

[Zhang *et al.*, 2015] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *NIPS*, pages 649–657, 2015.

[Zhao *et al.*, 2015] Han Zhao, Zhengdong Lu, and Pascal Poupart. Self-adaptive hierarchical sentence model. In *IJCAI*, pages 4069–4076, 2015.

[Zhao *et al.*, 2018] W Zhao, J Ye, M Yang, Z Lei, S Zhang, and Z Zhao. Investigating capsule networks with dynamic routing for text classification. In *EMNLP*, 2018.

[Zheng *et al.*, 2017] Zibin Zheng, Yatao Yang, et al. Wide and deep convolutional neural networks for electricity-theft detection to secure smart grids. *IEEE TII*, 2017.

[Zhou *et al.*, 2015] Chunting Zhou, Chonglin Sun, Zhiyuan Liu, and Francis Lau. A c-lstm neural network for text classification. *arXiv preprint arXiv:1511.08630*, 2015.