ACT: an Attentive Convolutional Transformer for Efficient Text Classification

Pengfei Li,[1] Peixiang Zhong,[1] Kezhi Mao,[1*]
Dongzhe Wang,[2] Xuefeng Yang,[2] Yunfeng Liu,[2] Jian-xiong Yin,[3] Simon SEE[3]
[1] Nanyang Technological University, Singapore
[2] ZhuiYi Technology, Shenzhen, China, [3] NVIDIA AI Tech Center
{pli006,peixiang001,ekzmao}@ntu.edu.sg, {ethanwang,ryan,glenliu}@wezhuiyi.com, {jianxiongy,ssee}@nvidia.com

Abstract

Recently, Transformer has been demonstrating promising performance in many NLP tasks and showing a trend of replacing Recurrent Neural Network (RNN). Meanwhile, less attention is drawn to Convolutional Neural Network (CNN) due to its weak ability in capturing sequential and long-distance dependencies, although it has excellent local feature extraction capability. In this paper, we introduce an Attentive Convolutional Transformer (ACT) that takes the advantages of both Transformer and CNN for efficient text classification. Specifically, we propose a novel attentive convolution mechanism that utilizes the semantic meaning of convolutional filters attentively to transform text from complex word space to a more informative convolutional filter space where important n-grams are captured. ACT is able to capture both local and global dependencies effectively while preserving sequential information. Experiments on various text classification tasks and detailed analyses show that ACT is a lightweight, fast, and effective universal text classifier, outperforming CNNs, RNNs, and attentive models including Transformer.

1 Introduction

Text classification is a fundamental problem behind many research topics in Natural Language Processing (NLP), such as topic categorization, sentiment analysis, relation extraction, etc. The key issue in text classification is text representation learning, which aims to capture both local and global dependencies of texts with respect to class labels. Compared with traditional bag-of-words/n-grams model (Wang and Manning 2012), deep neural networks have shown to be more effective since word order information can be utilized and more semantic features can be captured. The commonly adopted neural architectures in deep neural networks include CNN, RNN, and Transformer.

CNN is a special feed-forward neural network with convolutional layers interleaved with pooling layers. For NLP, the convolutional kernels/filters in CNN can be treated as n-gram extractors that convert n-gram in each position into a vector showing its relevance to the filters. With the help of pooling operations, the overall relevance of the text to each filter can be captured. Therefore, CNN has advantages in capturing semantic and syntactic information of n-grams for more abstract and discriminative representations (Kim 2014; Zhang, Zhao, and LeCun 2015; Li and Mao 2019). However, CNN is relatively weak in capturing sequential information and long-distance dependencies because convolutional filters normally have small kernel sizes focusing only on local n-grams, and the pooling operation results in loss of position information. Although we could apply dilated CNN (Yu and Koltun 2015) or construct deep CNNs with one layer stack on another to widen the convolution context to some extent, the performance gain is normally marginal with the cost of more data needed (Le, Cerisara, and Denis 2018). Besides, the convolutional filters in CNN may misfit to task-irrelevant words, hence producing non-discriminative features in the feature map (Li et al. 2017, 2020).

RNN is well-known for processing sequential data recurrently and it is widely used for text classification (Tang, Qin, and Liu 2015; Yogatama et al. 2017; Zhang et al. 2017a). However, RNN suffers from two problems due to its recurrent nature: gradient vanishing and parallel-unfriendly. Many works attempt to alleviate the gradient vanishing problem by incorporating attention mechanisms to RNN (Zhou et al. 2016; Yang et al. 2016; Zhang et al. 2017b). A novel neural architecture called Transformer (Vaswani et al. 2017) addresses both problems by relying entirely on self-attention to handle long-distance dependencies without recurrent computations. The emerging of Transformer-based neural networks has led to a series of breakthroughs in a wide range of NLP tasks (Zhang et al. 2018; Li et al. 2019; Zhong, Wang, and Miao 2019). Especially, the pre-trained language models based on Transformer have achieved state-of-the-art performance in many benchmark datasets (Devlin et al. 2019; Radford et al. 2019; Raffel et al. 2020). However, the heavy architecture of Transformer often requires more training data, CPU/GPU memory, and computational power, especially for long texts. Besides, since self-attention takes into account all the elements with a weighted averaging operation that disperses the attention distribution, Transformer may overlook the relation of neighboring elements (i.e. n-grams) that are important for text classification tasks (Yang et al. 2018, 2019a; Guo, Zhang, and Liu 2019).

To address the above-mentioned limitations of CNN and Transformer, we propose an Attentive Convolutional Transformer (ACT) which takes the advantages of both Trans-

[*] Corresponding author.

former and CNN for efficient text classification. Similar as Transformer, ACT also has a multi-head structure that jointly performs attention operations in different subspaces. However, instead of self-attention, a novel attentive convolution mechanism is performed in each attention head to better capture local n-gram features. Different from conventional CNN, the proposed attentive convolution utilizes the semantic meaning of convolutional filters attentively and transforms texts from complex word space to a more informative convolutional filter space. This not only simplifies the optimization of capturing important n-grams for classification, but also allows our model to learn meaningful convolutional filters since all the filters contribute to the final representation directly. Compared with self-attention, the proposed attentive convolution focuses more on learning important local n-gram features globally which are invariant to the specific inputs. These n-gram features are exactly the keywords and phrases that are crucial for text classification. While majority of existing works augment Transformer with conventional CNNs to improve locality modeling capability with the cost of introducing more parameters (Yu et al. 2018; Mohamed, Okhonko, and Zettlemoyer 2019; Yang et al. 2019a; Gulati et al. 2020), our work is a more lightweight approach and it is the first to utilize the semantic meaning of convolutional filters with attention mechanism.

The proposed ACT is also sequence-to-sequence, with an additional global representation output by keeping the max-pooling functionality of CNN. Therefore, it is able to capture both local and global features while preserving sequential information. Furthermore, we propose a global attention mechanism to summarize the outputs of ACT and obtain the final representation by taking local, global, and position information into consideration. Experiments are conducted on typical text classification tasks including sentiment analysis and topic categorization, as well as the more challenging relation extraction task. We present detailed analyses on ACT, results show that ACT is a lightweight and efficient universal text classifier, outperforming existing CNN-based, RNN-based, and attentive models including Transformer.

2 Attentive Convolutional Transformer

We present the proposed ACT in detail in this section. The attentive convolution mechanism of ACT is introduced in Section 2.1; the multi-head multi-layer structure of ACT is described in Section 2.2; the global attention mechanism for final text representation is presented in Section 2.3.

2.1 Attentive Convolution Mechanism

Attentive convolution mechanism is the fundamental operation of ACT. It first performs n-gram convolution over text, then transforms text into convolutional filter space by combining the filters attentively. With different utilization of feature maps as attention weights, attentive convolution mechanism is able to capture both local and global features of texts. The architecture of the proposed attentive convolution is shown in Figure 1 (left).

Local feature representation Given a text input $t = [t_1, t_2, ..., t_l]$, we first represent each word token t_i as word embedding $\mathbf{q}_i \in \mathbb{R}^{d_w}$ and obtain the input embeddings $\mathbf{Q} = [\mathbf{q}_1, \mathbf{q}_2, ..., \mathbf{q}_l]$ by looking up the word embedding matrix $\mathbf{W}^{wrd} \in \mathbb{R}^{d_w \times V}$, where d_w is the dimension of word embedding and V is vocabulary size. Then, n-gram convolution over input embeddings $\mathbf{Q}$ is performed using convolutional filters $\mathbf{F} = [\mathbf{f_1}, \mathbf{f_2}, ..., \mathbf{f_m}]$, where $\mathbf{f_i} \in \mathbb{R}^{nd_w}$ and n is the convolution kernel size. A feature map matrix $\mathbf{M} \in \mathbb{R}^{m \times l}$ is generated as follows:

$$\mathbf{M} = \mathbf{Q} \circledast \mathbf{F} \tag{1}$$

where $\circledast$ indicates the convolution operation of $\mathbf{f_i}$ over $\mathbf{Q}$. Specifically, the value in the feature map is calculated as shown in Equation 2:

$$m_{ij} = f({\mathbf{f_i}}^T \cdot \mathrm{Cat}(q_j, q_{j+1}, ..., q_{j+n-1}) + b) \tag{2}$$

where Cat means concatenation, f is a non-linear activation function and b is a bias term.

The values in the resulted feature map indicate semantic relevance between n-grams and convolutional filters. By treating the feature map values as attention weights and aggregating the semantic convolutional filters attentively, we transform each n-gram from complex word space to a more informative convolutional filter space while preserving the sequential information of texts. Formally, the attentive convolution for local feature representation is shown in Equation 3.

$$\mathbf{O} = \mathbf{F} \cdot \mathbf{M} = \mathbf{F} \cdot (\mathbf{Q} \circledast \mathbf{F}) \tag{3}$$

where $\mathbf{O} = [\mathbf{o}_1, \mathbf{o}_2, ..., \mathbf{o}_l] \in \mathbb{R}^{nd_w \times l}$ is the output obtained from attentive convolution.

Different from self-attention whose output space is still a complex word space with varying components depending on the input, the output space in our proposed attentive convolution mechanism is formed by n-gram convolutional filters which are learned globally and invariant to the inputs. In such space, important n-grams will be close to the corresponding filters and irrelevant n-grams will have small values. Therefore, the important local features (n-grams) appear in the texts can be captured effectively.

Global feature representation Besides local features, attentive convolution mechanism can also capture global features of texts by applying the max-pooling technique which is normally used in conventional CNNs. The max-pooling over each row of the feature map $\mathbf{M}$ finds the overall relevance of the texts to each convolutional filter. By aggregating the convolutional filters attentively using the max-pooling results, we can find the overall semantics of texts in the filter space. Formally, the attentive convolution for global feature representation is shown in Equation 4.

$$\mathbf{g} = \mathbf{F} \cdot max(\mathbf{M}) \tag{4}$$

where $\mathbf{g} \in \mathbb{R}^{nd_w}$ and max means row-wise max-pooling.

Comparison with existing methods Compared with conventional CNN whose outputs come from feature maps only, our proposed attentive convolution utilizes both feature maps and semantic meaning of convolutional filters for text representation. This allows our model to learn meaningful convolutional filters effectively since all the filters contribute

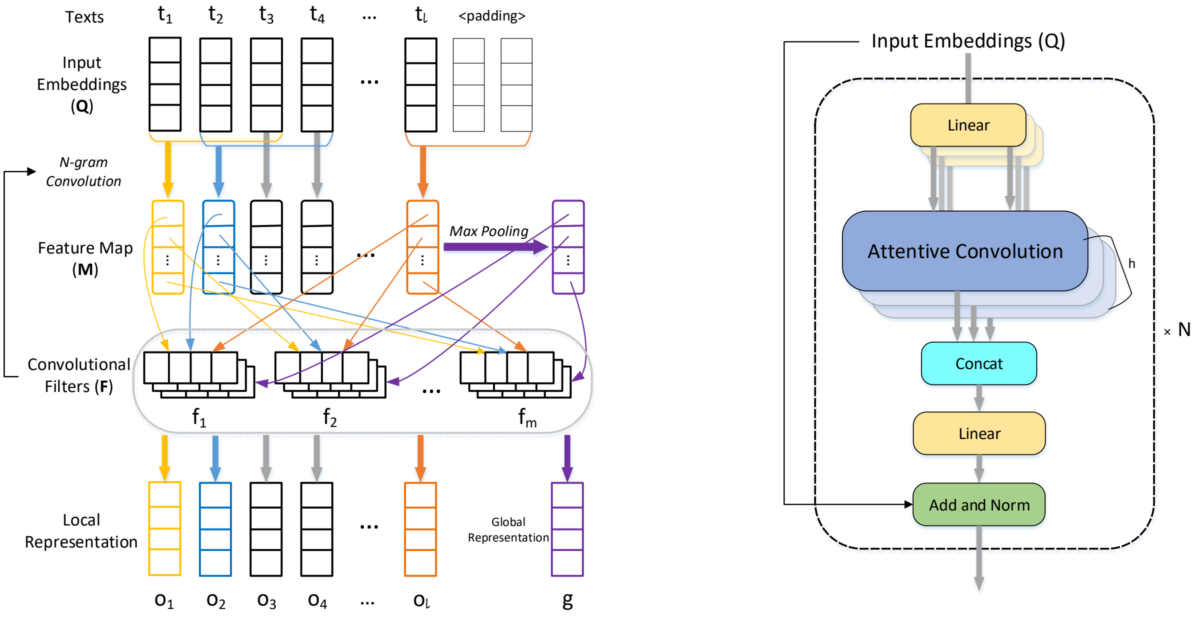

Figure 1: Left: attentive convolution mechanism. Outputs are obtained by combining convolutional filters attentively utilizing feature map as attention weights. Right: multi-head multi-layer structure of ACT. h and N indicate number of attentive convolution heads and layers respectively.

to the final representation directly. Moreover, the pooling operation in conventional CNN ignores the sequential information of texts, whereas the local feature representation in our method preserves the sequential information while capturing important n-gram features.

Compared with conventional attention mechanism whose attention weights are calculated from vector product of queries ($\mathbf{Q}$) and keys ($\mathbf{K}$), our proposed method calculates attention weights through convolution of queries ($\mathbf{Q}$) using the keys ($\mathbf{F}$), where the keys and values in our attention mechanism are convolutional filters learned during end-to-end training. The convolution operation involves wider context (n-grams) than the vector product of single words, this allows our model to capture important n-gram features more effectively. These n-gram features are exactly the keywords and phrases that are crucial for text classification. Besides, as mentioned in Section 2.1, the output space is more simplified and informative since it is formed by convolutional filters that are invariant to the inputs.

2.2 Multi-head Multi-layer Attentive Convolution

Inspired by Transformer (Vaswani et al. 2017), the proposed ACT also has multi-head and multi-layer structures as shown in Figure 1 (right).

For h-head ACT, we first linearly transform input embeddings $\mathbf{Q}$ h times and perform h attentive convolution simultaneously. Then the outputs from different attention heads are concatenated together and linearly transformed to the original input dimension, as shown in Equation 5.

$$\begin{aligned}\text{MultiHead}(\mathbf{Q}) &= \mathbf{W}^O\text{Cat}(\mathbf{O}_1, \mathbf{O}_2, ..., \mathbf{O}_h)\\ \text{where } \mathbf{O}_i &= \text{AttenConv}(\mathbf{W}_i^Q\mathbf{Q})\end{aligned} \tag{5}$$

Here, AttenConv indicates the proposed attentive convolution mechanism, $\mathbf{W}_i^Q \in \mathbb{R}^{(d_w/h)\times d_w}$ and $\mathbf{W}^O \in \mathbb{R}^{d_w \times n d_w}$ are the weight matrices of linear transformations. Furthermore, we adopt the residual connection and layer norm as used in Vaswani et al. (2017). For multi-layer ACT, we simply pass the local representations of lower-layer to the input of upper-layer to obtain the higher-level local representations. The global representation is obtained from the top ACT layer.

The multi-head structure of ACT allows our model to jointly capture important n-gram features in different subword spaces, where the n-grams in different spaces have different contributions to the final representation. The multi-layer structure allows our model to capture higher-level semantics effectively. Since the upper-layer involves a wider context for convolution, it is able to induce more abstract and discriminative representations.

2.3 Global Attention and Classification

To obtain the final representation of texts for classification, we propose a global attention mechanism that summarizes the sequential outputs of ACT. As shown in Figure 2, the attention weights are calculated by taking both local and global representations as well as position information of each token into consideration.

The local representation $\mathbf{O} \in \mathbb{R}^{d_w \times l}$ and global representation $\mathbf{g} \in \mathbb{R}^{d_w}$ are obtained from the top-layer of ACT. The position embedding $\mathbf{P} \in \mathbb{R}^{d_p \times l}$ is obtained by mapping each token's absolute position to d_p-dimensional embeddings based on a trainable position embedding matrix $\mathbf{W}^p \in \mathbb{R}^{d_p \times P}$, where P is the total number of positions.

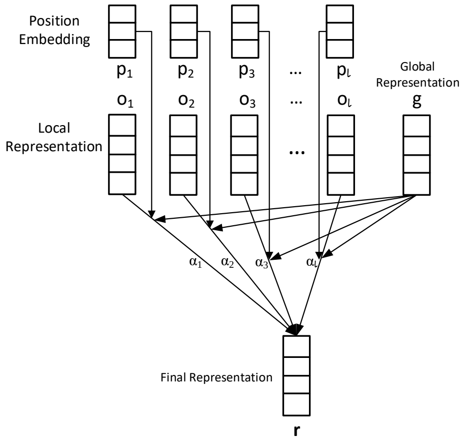

Figure 2: Global attention mechanism. Attention weights α_i are calculated based on local representation $\mathbf{o}_i$, global representation $\mathbf{g}$, and position embedding $\mathbf{p}_i$ of each token.

The final text representation is obtained by Equation 6:

$$\mathbf{r} = \mathbf{O} \cdot \text{Softmax}(f(\mathbf{W}_o\mathbf{O} + \mathbf{W}_p\mathbf{P})^T\mathbf{c} + \frac{\mathbf{O}^T\mathbf{g}}{\sqrt{d_w}}) \qquad (6)$$

where f is a non-linear activation function, $\mathbf{W}_o \in \mathbb{R}^{d_a \times d_w}$ $\mathbf{W}_p \in \mathbb{R}^{d_a \times d_p}$ are linear transformation weight matrices, d_a is attention dimension, $\mathbf{c} \in \mathbb{R}^{d_a}$ is a context vector learned by the neural network, $\sqrt{d_w}$ is a scaling factor depends on input dimension.

For classification, we pass the final representation $\mathbf{r}$ to a classifier consisting of a fully connected layer and a softmax layer to predict class probabilities. Our model is trained by minimizing categorical cross-entropy loss and center loss (Wen et al. 2016) using stochastic gradient descent (SGD) with momentum and learning rate decay.

3 Experiments

We evaluate our proposed ACT on three different text classification tasks, including sentiment analysis, topic categorization, and relation extraction. Since relation extraction is slightly different from traditional text classification tasks where special considerations are needed for target entities, we conduct experiments on it separately.

3.1 Datasets

We use six widely-studied datasets to evaluate our model, two for each text classification task. These datasets are diverse in the aspects of type, size, number of classes, and document length. Table 1 shows the statistics of the datasets.

For sentiment analysis, we use two datasets constructed by Zhang et al. (2015) which are obtained from Yelp Dataset Challenge 2015. Yelp Review Polarity (Yelp P.) is a binary sentiment classification dataset whose class is either positive or negative; Yelp Review Full (Yelp F.) contains more fine-grained sentiment classes ranging from rating 1 to 5.

For topic categorization, we use AG's News (AGNews) and DBPedia datasets created by Zhang et al. (2015). AGNews contains news articles from four categories: world, entertainment, sports, and business; DBPedia is an ontology classification dataset containing 14 non-overlapping categories picked from DBpedia 2014.

For relation extraction, we use TACRED and SemEval2010-task8 (SemEval) datasets which contain hand-annotated subject and object entities as well as the relation type between the entities. TACRED is a large-scale and complex relation extraction dataset constructed by Zhang et al. (2017b) which has 41 relation types and a *no_relation* class; SemEval2010-task8 (Hendrickx et al. 2009) is a relatively smaller relation extraction dataset which has 9 directed relations and 1 other relation.

3.2 Baseline Models

A variety of baseline models are used for comparison with our model. Different baseline models are used for relation extraction since the task is more challenging and normally requires dedicated models.

Text classification models

- **CNN-based models** including Word-level CNN, Char-level CNN (Zhang, Zhao, and LeCun 2015), and deep CNN namely VDCNN (Conneau et al. 2016).
- **RNN-based models** including standard LSTM (Zhang, Zhao, and LeCun 2015), discriminative LSTM (D-LSTM) of Yogatama et al. (2017), and Skim-LSTM which dynamically updates its hidden states (Seo et al. 2018).
- **Attentive models** including bi-directional block self-attention network (Bi-BloSAN) (Shen et al. 2018), label-embedding attentive model (LEAM) (Wang et al. 2018), and Transformer encoder (Vaswani et al. 2017) for text classification.

Relation extraction models

- **CNN-based models** including the standard CNN for sentence classification (Kim 2014), CNN with position embeddings (CNN-PE) (Nguyen and Grishman 2015), and graph convolutional network (GCN) over pruned dependency trees of sentences (Zhang, Qi, and Manning 2018).
- **RNN-based models** including standard LSTM and LSTM with position-aware attention (PA-LSTM) (Zhang et al. 2017b).
- **CNN-RNN hybrid model** including contextualized GCN (C-GCN) where the input vectors are obtained using bi-LSTM (Zhang, Qi, and Manning 2018).
- **Attentive models** including Transformer encoder (Bilan and Roth 2018), knowledge-attention encoder (Knwl-attn) (Li et al. 2019), and knowledge-attention self-attention integrated model (Knwl+Self).

3.3 Experiment Settings

In our experiments, word embedding matrix $\mathbf{W}^{wrd}$ is initialized with 300-d Glove word embeddings (Pennington, Socher, and Manning 2014). The fully connected layer before softmax has a dimension of 100. Dropout regularization (Srivastava et al. 2014) with a rate of 0.4 is applied

Datasets	Types	Classes	Average lengths	Train samples	Test samples
Yelp Review Polarity (Yelp P.)	Sentiment	2	156	560,000	38,000
Yelp Review Full (Yelp F.)	Sentiment	5	158	650,000	50,000
AG's News (AGNews)	Topic	4	44	120,000	7,600
DBPedia	Topic	14	55	560,000	70,000
TACRED	Relation	41	36	90,755	15,509
SemEval2010-task8 (SemEval)	Relation	19	19	8,000	2,717

Table 1: Statistics of the six text classification datasets used in our experiments.

Model	Yelp P.	Yelp F.	AGNews	DBPedia
Word-level CNN	95.40	59.84	91.45	98.58
Char-level CNN	94.75	61.60	90.15	98.34
VDCNN	95.72	64.72	91.33	98.71
LSTM	94.74	58.17	86.06	98.55
D-LSTM	92.60	59.60	92.10	98.70
Skim-LSTM	/	/	93.60	/
Bi-BloSAN	94.56	62.13	93.32	98.77
LEAM	95.31	64.09	92.45	99.02
Transformer	96.13*	65.34*	93.89*	98.98*
ACT	**97.41**	**68.16**	**94.25**	**99.19**

Model	TACRED	SemEval
CNN	59.3*	70.0*
CNN-PE	61.4*	82.3*
GCN	64.0	/
LSTM	61.5*	80.9*
PA-LSTM	65.1	82.7
C-GCN	66.4	**84.8**
Knwl-attn	66.4	82.3
Knwl+Self	67.8	84.3
Transformer	66.5	83.1
ACT	**68.0**	84.5

Table 2: Left: classification accuracy (%) on sentiment analysis and topic categorization tasks. Right: F_1 scores on relation extraction task, official micro-averaged and macro-averaged F_1 scores are used for TACRED and SemEval2010-task8 datasets respectively. * means the results are obtained from our implementation. / means not reported. All other results are directly cited from the respective papers mentioned in Section 3.2.

during training. The weight and learning rate for center loss are 0.001 and 0.1 respectively. The models are trained using SGD with initial learning rate of 0.01 and momentum of 0.9. Learning rate is decayed with a rate of 0.9 after 10 epochs if the score on the development set does not improve. Batch size is set to 100 and the model is trained for 70 epochs. The dimensions of global attention and position embedding are 200 and 60 respectively. We use GeLUs (Hendrycks and Gimpel 2016) for all the nonlinear activation functions.

The hyper-parameters of ACT are selected by grid-search (refer to Section 4.2 for details). Specifically, for sentiment analysis and topic categorization, we set aside 10% of training data as the development set to tune model hyper-parameters. We report the average classification accuracy on the test set based on 5 independent runs. For ACT, we use 3-layer encoder with 6 attentive convolution heads in each layer, and $m = 100$ convolutional filters with a kernel size of 3 in the attentive convolution mechanism. For relation extraction, we use the same settings as Zhang et al. (2017b) for a fair comparison with baseline models. Particularly, instead of using absolute positions in global attention, we use two relative positions for each token with respect to the two target entities. Each relative position embedding has a dimension of 30 and they are concatenated together as final position embedding. For ACT, we use one layer encoder with 6 attentive convolution heads in each layer, and $m = 40$ convolutional filters with a kernel size of 3 in the attentive convolution mechanism.

3.4 Results and Analysis

Experiment results on the six text classification datasets are shown in Table 2. Left table shows the classification accuracy on sentiment analysis and topic categorization tasks; right table shows the F_1 score on relation extraction task. Our proposed ACT achieves the best performance among all the baseline models for majority of datasets. For SemEval dataset, ACT ranks the 2nd best and has comparable performance with C-GCN, a sophisticated model for relation extraction.

Compared with CNN-based models, ACT performs better than shallow CNN (word/character-level), graph convolution network (GCN), and deep CNN (VDCNN) with a significant margin. The reason is that ACT is able to capture both local n-gram features and global dependencies effectively while preserving sequential information. Besides, the learning of convolutional filters is more efficient using the proposed attentive convolution mechanism where semantic meanings of the filters are utilized for text representation.

Compared with RNN-based models, ACT consistently outperforms standard LSTM and improved variants of LSTM (D-LSTM, Skim-LSTM, and PA-LSTM) for all the tasks. This credits to the attentive convolution mechanism for better capturing n-gram features, as well as the multi-head multi-layer structure that does not suffer from gradient vanishing problem when capturing long-distance dependencies. The contextualized GCN (C-GCN) using bi-LSTM and GCN performs slightly better than ACT on SemEval dataset,

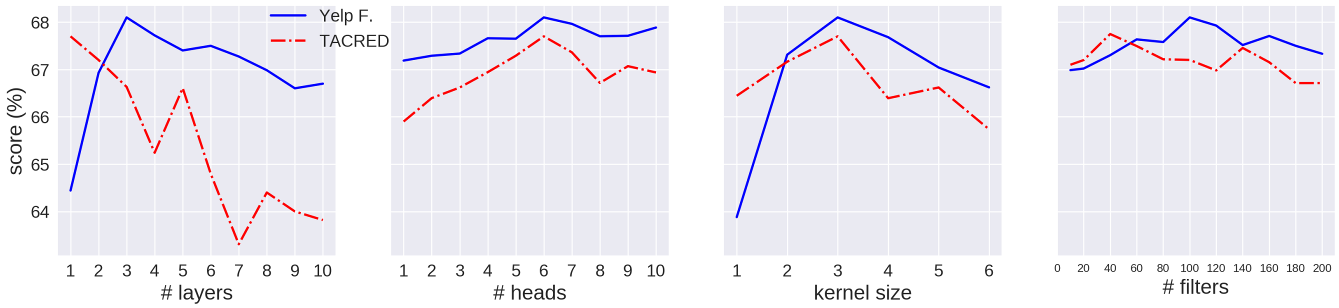

Figure 3: Hyper-parameter study on ACT. X-coordinate indicates the hyper-parameters studied, Y-coordinate indicates classification accuracy for Yelp F. dataset and micro-averaged F_1 score for TACRED dataset.

probably due to the benefits of dependency trees. Our model does not require any dependency parsing of the sentences.

It is observed that attentive models generally outperform RNN-based models. This is due to the better ability of attention mechanisms in capturing long-distance dependencies, especially the self-attention used in Transformer. The proposed ACT outperforms all the attentive models including Transformer encoder. The reason is that ACT has better local n-gram feature extraction capability by using attentive convolution mechanism. However, important n-gram features may not be captured effectively by Transformer because each token will attend to the whole sequence instead of n-grams, the output may be affected by irrelevant tokens. Besides, ACT also simplifies the optimization because it transforms text representation from complex word space to more informative filter space, leading to more stable training and better keyword extraction capability.

The recently proposed knowledge-attention and self-attention integrated model (Li et al. 2019) performs as well as ACT on relation extraction task, with the aid of external lexical resources to better capture the keywords of relations. Encouragingly, our proposed ACT is able to capture such keywords effectively without the need of external knowledge resources, yet achieving better performance.

4 Discussions

We present more in-depth analyses and discussions on ACT in this section. Two relatively different datasets are used to conduct our experiments: one is Yelp F., a large dataset for sentiment analysis; the other is TACRED, a relation extraction dataset which is much smaller. We report accuracy and micro-averaged F_1 score on the development sets of Yelp F. and TACRED respectively.

4.1 Ablation Study

We perform an ablation study on ACT to investigate the contributions of specific components of ACT. Results are shown in Table 3.

(1) We replace the proposed attentive convolution mechanism with conventional CNN where feature maps are used for text representation directly, the performance drop by 1.8-1.9%. This demonstrates the advantage of utilizing the semantic meaning of convolutional filters attentively for text representation. (2) The proposed multi-head structure outperforms single-head significantly, showing the effectiveness of jointly capturing n-gram features in different subword spaces in the multi-head structure. (3) Removing the global representation in global attention degrades the performance by 1%. This demonstrates that incorporating global representation into the attention mechanism yields better attention weights for local representations. (4) After removing the position embeddings in global attention, the performance drops by 1.3% for Yelp F. and 6.3% for TACRED. This shows that position information is important for text classification, especially for relation extraction task.

Model	Yelp F.	TACRED
ACT	68.3	67.8
1. − Attentive Conv.	67.1	66.5
2. − Multi-head	67.0	65.9
3. − Global rep.	67.6	67.1
4. − Position embed.	67.4	63.5

Table 3: Ablation study on ACT. Accuracy (%) and micro-averaged F_1 score are reported on the development sets of Yelp F. and TACRED respectively.

4.2 Hyper-parameter Study

In this section, we study the influence of some important hyper-parameters on the performance of ACT, including number of layers, number of attentive convolution heads, kernel size, and number of filters in attentive convolution. Experiment results are shown in Figure 3.

It is observed that the number of ACT layers affects the performance significantly. For small datasets like TACRED, single-layer ACT achieves the best performance. For large datasets like Yelp F., the optimal number of layers is 3. Further increasing the number of layers will increase model complexity and cause performance drop due to overfitting. Besides, multiple attentive convolution heads are beneficial for ACT, the optimal number of heads is 6. For kernel size, results show that 3-gram convolution is most effective for ACT.[1] It is also observed that ACT is not very

[1] We also tried using multiple kernel sizes simultaneously, results show no improvements over single kernel size.

Sample Sentences	True Class	Prediction
OBJ-PERSON returned to Buffalo in 1955 and was a part of a group of black intellectuals who included philosopher and poet SUBJ-PERSON SUBJ-PERSON , whom she married in 1958 .	spouse	no relation
OBJ-PERSON returned to Buffalo in 1955 and was a part of a group of black intellectuals who included philosopher and poet SUBJ-PERSON SUBJ-PERSON , whom she married in 1958 .		spouse
When I worked at the Renaissance tower , I 'd come here when I was too lazy to walk down the street for something better . Because , honestly , their pizza just is n't that great . Or good , really . But I 've had the breakfast muffin twice and both times it was beyond awesome ! Just the right amount of grease to let you know it 's good . And super cheap !	3 star	5 star
When I worked at the Renaissance tower , I 'd come here when I was too lazy to walk down the street for something better . Because , honestly , their pizza just is n't that great . Or good , really . But I 've had the breakfast muffin twice and both times it was beyond awesome ! Just the right amount of grease to let you know it 's good . And super cheap !		3 star

Table 4: Attention visualization for Transformer and ACT. For each sample, the visualization of Transformer is presented first, followed by our proposed ACT. Words are highlighted based on the attention weights assigned to them. Best viewed in color.

sensitive to number of convolutional filters. However, larger dataset (Yelp F.) requires more filters than smaller dataset (TACRED) to achieve the best performance.

4.3 Attention Visualization

To investigate what ACT focuses on, as well as its difference from Transformer, we conduct visualization of attention weights assigned to words. We sample sentences from the development sets of Yelp F. and TACRED. The visualization results are shown in Table 4. More visualizations are presented in the Appendix due to space limitation.

The visualization results show that the proposed ACT can capture the keywords and cue phrases more effectively than Transformer. It is observed that Transformer attends to a wide range of words in the sentence, including stop words and punctuations which may be irrelevant for the classification task. On the contrary, ACT pays more attention to the important n-grams such as “married” for “spouse” relation and “is n’t” for fine-grained sentiment classification. These n-grams are the keywords and cue phrases of certain class which are crucial for classification tasks.

4.4 Model Size and Inference Speed

In this section, we investigate two practical aspects of our model for real-world applications: model size and inference speed. For model size, we report the number of model parameters. For inference speed, we report the average time needed to compute a single batch (batch size of 100) of Yelp F. dataset using NVIDIA Tesla P40 GPU with Intel Xeon E5-2667 CPU. We also compare our model with Transformer under the same hyper-parameter settings as described in Section 3.3, results are shown in Table 5.

Model	# param.	Inf. time
Transformer	3.38M	0.19s
ACT	1.49M	0.07s

Table 5: Comparison of model parameters and inference time per batch on Yelp F. dataset.

The proposed ACT is much smaller and faster compared with Transformer. It has 56% fewer parameters and 2.7 times faster inference speed. Therefore, ACT is a light-weight and efficient model for text classification, and it is more practical for real-world applications. Although the large pre-trained language models based on Transformer such as BERT (Devlin et al. 2019) and XLNet (Yang et al. 2019b) have achieved start-of-the-art performance in many NLP tasks, the memory and speed constraints will become obstacles for practical applications.

5 Conclusion and Future Work

We introduce an Attentive Convolutional Transformer (ACT) for efficient text classification. By taking the advantages of both Transformer and CNN, ACT is able to capture both local and global dependencies effectively while preserving sequential information of texts. Particularly, a novel attentive convolution mechanism is proposed to better capture n-gram features in convolutional filter space. We also propose a global attention mechanism to obtain the final representation by taking local, global, and position information into consideration. Detailed analyses show that ACT is a lightweight and efficient universal text classifier that achieves consistently good results over different text classification tasks, outperforming CNN-based, RNN-based, and attentive models including Transformer.

Although our proposed ACT is dedicated for text classification tasks where local feature extraction capability is of particular importance, we will explore the potential applications of ACT on other NLP tasks such as machine translation, text summarization, and language modeling in future work. Furthermore, we will apply the idea of the proposed attentive convolution mechanism to other fields beyond NLP domain, such as speech recognition and computer vision.

References

Bilan, I.; and Roth, B. 2018. Position-aware Self-attention with Relative Positional Encodings for Slot Filling. *arXiv preprint arXiv:1807.03052* .

Conneau, A.; Schwenk, H.; Barrault, L.; and Lecun, Y. 2016. Very deep convolutional networks for natural language processing. *arXiv preprint arXiv:1606.01781* 2.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 4171–4186.

Gulati, A.; Qin, J.; Chiu, C.-C.; Parmar, N.; Zhang, Y.; Yu, J.; Han, W.; Wang, S.; Zhang, Z.; Wu, Y.; et al. 2020. Conformer: Convolution-augmented Transformer for Speech Recognition. *arXiv preprint arXiv:2005.08100* .

Guo, M.; Zhang, Y.; and Liu, T. 2019. Gaussian transformer: a lightweight approach for natural language inference. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 6489–6496.

Hendrickx, I.; Kim, S. N.; Kozareva, Z.; Nakov, P.; Ó Séaghdha, D.; Padó, S.; Pennacchiotti, M.; Romano, L.; and Szpakowicz, S. 2009. Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. In *Proceedings of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions*, 94–99. Association for Computational Linguistics.

Hendrycks, D.; and Gimpel, K. 2016. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415* .

Kim, Y. 2014. Convolutional Neural Networks for Sentence Classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1746–1751.

Le, H. T.; Cerisara, C.; and Denis, A. 2018. Do convolutional networks need to be deep for text classification? In *Workshops at the Thirty-Second AAAI Conference on Artificial Intelligence*.

Li, H.; Kadav, A.; Durdanovic, I.; Samet, H.; and Graf, H. P. 2017. Pruning Filters for Efficient ConvNets. In *5th International Conference on Learning Representations, ICLR 2017*.

Li, P.; and Mao, K. 2019. Knowledge-oriented convolutional neural network for causal relation extraction from natural language texts. *Expert Systems with Applications* 115: 512–523.

Li, P.; Mao, K.; Yang, X.; and Li, Q. 2019. Improving Relation Extraction with Knowledge-attention. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 229–239.

Li, Q.; Li, P.; Mao, K.; and Lo, E. Y.-M. 2020. Improving convolutional neural network for text classification by recursive data pruning. *Neurocomputing* 414: 143–152.

Mohamed, A.; Okhonko, D.; and Zettlemoyer, L. 2019. Transformers with convolutional context for ASR. *arXiv preprint arXiv:1904.11660* .

Nguyen, T. H.; and Grishman, R. 2015. Relation extraction: Perspective from convolutional neural networks. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, 39–48.

Pennington, J.; Socher, R.; and Manning, C. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 1532–1543.

Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; and Sutskever, I. 2019. Language models are unsupervised multitask learners. *OpenAI Blog* 1(8): 9.

Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; and Liu, P. J. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research* 21(140): 1–67.

Seo, M.; Min, S.; Farhadi, A.; and Hajishirzi, H. 2018. Neural speed reading via skim-rnn. *International Conference on Learning Representations* .

Shen, T.; Zhou, T.; Long, G.; Jiang, J.; and Zhang, C. 2018. Bi-directional block self-attention for fast and memory-efficient sequence modeling. In *International Conference on Representation Learning*.

Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; and Salakhutdinov, R. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research* 15(1): 1929–1958.

Tang, D.; Qin, B.; and Liu, T. 2015. Document modeling with gated recurrent neural network for sentiment classification. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, 1422–1432.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in neural information processing systems*, 5998–6008.

Wang, G.; Li, C.; Wang, W.; Zhang, Y.; Shen, D.; Zhang, X.; Henao, R.; and Carin, L. 2018. Joint Embedding of Words and Labels for Text Classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2321–2331.

Wang, S.; and Manning, C. D. 2012. Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the 50th annual meeting of the association for computational linguistics: Short papers-volume 2*, 90–94. Association for Computational Linguistics.

Wen, Y.; Zhang, K.; Li, Z.; and Qiao, Y. 2016. A discriminative feature learning approach for deep face recognition. In *European conference on computer vision*, 499–515. Springer.

Yang, B.; Tu, Z.; Wong, D. F.; Meng, F.; Chao, L. S.; and Zhang, T. 2018. Modeling Localness for Self-Attention Networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 4449–4458.

Yang, B.; Wang, L.; Wong, D. F.; Chao, L. S.; and Tu, Z. 2019a. Convolutional Self-Attention Networks. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 4040–4045.

Yang, Z.; Dai, Z.; Yang, Y.; Carbonell, J.; Salakhutdinov, R.; and Le, Q. V. 2019b. XLNet: Generalized Autoregressive Pretraining for Language Understanding. *arXiv preprint arXiv:1906.08237* .

Yang, Z.; Yang, D.; Dyer, C.; He, X.; Smola, A.; and Hovy, E. 2016. Hierarchical Attention Networks for Document Classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 1480–1489. San Diego, California: Association for Computational Linguistics. doi:10.18653/v1/N16-1174. URL https://www.aclweb.org/anthology/N16-1174.

Yogatama, D.; Dyer, C.; Ling, W.; and Blunsom, P. 2017. Generative and discriminative text classification with recurrent neural networks. In *Thirty-fourth International Conference on Machine Learning (ICML 2017)*. International Machine Learning Society.

Yu, A. W.; Dohan, D.; Luong, M.-T.; Zhao, R.; Chen, K.; Norouzi, M.; and Le, Q. V. 2018. QANet: Combining Local Convolution with Global Self-Attention for Reading Comprehension. In *International Conference on Learning Representations*.

Yu, F.; and Koltun, V. 2015. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122* .

Zhang, H.; Xiao, L.; Wang, Y.; and Jin, Y. 2017a. A generalized recurrent neural architecture for text classification with multi-task learning. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, 3385–3391. AAAI Press.

Zhang, J.; Luan, H.; Sun, M.; Zhai, F.; Xu, J.; Zhang, M.; and Liu, Y. 2018. Improving the Transformer Translation Model with Document-Level Context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 533–542.

Zhang, X.; Zhao, J.; and LeCun, Y. 2015. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, 649–657.

Zhang, Y.; Qi, P.; and Manning, C. D. 2018. Graph Convolution over Pruned Dependency Trees Improves Relation Extraction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2205–2215.

Zhang, Y.; Zhong, V.; Chen, D.; Angeli, G.; and Manning, C. D. 2017b. Position-aware attention and supervised data improve slot filling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 35–45.

Zhong, P.; Wang, D.; and Miao, C. 2019. Knowledge-Enriched Transformer for Emotion Detection in Textual Conversations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 165–176.

Zhou, P.; Shi, W.; Tian, J.; Qi, Z.; Li, B.; Hao, H.; and Xu, B. 2016. Attention-based bidirectional long short-term memory networks for relation classification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, 207–212.