Sample-adaptive Classification Inference Network

Juan Yang (yangjuan6985@163.com)
Hefei University of Technology

Guanghong Zhou
Hefei University of Technology

Ronggui Wang
Hefei University of Technology

Lixia Xue
Hefei University of Technology

Research Article

Keywords: Adaptive, Sample difference, Exit mechanism, Feature enhancement

Posted Date: March 24th, 2023

DOI: https://doi.org/10.21203/rs.3.rs-2718558/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.
Sample-adaptive Classification Inference Network

Juan Yang¹*, Guanghong Zhou¹, Ronggui Wang¹ and Lixia Xue¹

¹*School of Computer and Information, Hefei University of Technology, Danxia Road, Hefei, 230601, Anhui, China.

*Corresponding author(s). E-mail(s): yangjuan6985@163.com; guanghong0108@163.com; wangrgui@foxmail.com; xlxzzm@163.com;

Abstract

Existing pre-trained models have yielded promising results in terms of computational time reduction. However, these models only focus on pruning simple sentences or less salient words, while neglecting the treatment of relatively complex sentences. It is frequently these sentences that cause the loss of model accuracy. This shows that the adaptation of the existing models is one-sided. To address this issue, in this paper, we propose a sample-adaptive training and inference model. Specifically, complex samples are extracted from the training datasets and a dedicated data augmentation module is trained to extract global and local semantic information of complex samples. During inference, simple samples can exit the model via the Sample Adaptive Exit Mechanism, Normal samples pass through the whole backbone model before inference, while complex samples are processed by the Characteristic Enhancement Module after passing through the backbone model. In this way, all samples are processed adaptively. Our extensive experiments on classification tasks datasets in the field of Natural Language Processing demonstrate that our method enhances model accuracy and reduces model inference time for multiple datasets. Moreover, our method is transferable and can be applied to multiple pre-trained models.

Keywords: Adaptive, Sample difference, Exit mechanism, Feature enhancement
1 Introduction

Classification tasks are essential tasks in Natural Language Processing (NLP), and pre-trained language models have shown impressive results on those tasks. Pre-trained models such as BERT [1], RoBERTa [2] can learn general knowledge from a large amount of unlabeled data and apply them to classification tasks with great success. However, these models are too large and computationally expensive for practical use [3][4], which limits their utility. To overcome this issue, various techniques such as knowledge distillation [5][6][7], shared weight mechanism [8][9], and token/sample adaptive mechanisms have been proposed [10][11][12]. While some methods have been proposed to reduce the computational cost of pre-trained models, they only consider simple samples or feature tokens while neglecting more complex ones. However, the difficulty of semantic analysis of human languages is different, and some sentences are more complex than others [13]. These complex samples are often poorly handled by the model and affect the overall performance of the model. To address this issue, we propose a sample-adaptive training and inference model that can extract complex samples from the training datasets and train a separate data augmentation module to extract their global and local semantic information. During inference, simple samples can be adaptively removed from the model using an exit mechanism [14], while complex samples continue to be processed through the trained data augmentation module. We conducted extensive experiments on classification tasks datasets in the field of NLP and found that our approach not only improves the accuracy of the model but also reduces the inference time on multiple datasets. Our method is also transferable and can be applied to various pre-trained models. Overall, our proposed approach provides a promising direction for optimizing pre-trained language models by taking into account the complexity of semantic analysis in natural language.

![Image](image.png)

Fig. 1 This is a case of text semantic analysis: The above figure shows the scores obtained by the model for simple and complex samples.

In summary, this paper proposes a more detailed sample adaptive training and inference model for classification tasks in the field of NLP, based on previous studies on pre-trained language models. The model includes a sample feature enhancement module that uses Transformer blocks and CNN structures to fuse global and local features of complex samples. As well as mechanisms for simple sample withdrawal and complex sample extraction and inference. Additionally, the paper presents a classifier based on the Attention on Attention structure (AOA) [15][16][17] to improve the correlation between output
and query. The method is implemented on multiple baseline and datasets and experiments show that it accelerates inference while improving accuracy. The contributions of the paper include the proposal of a more detailed adaptive sample training and inference model, firstly use of AOA structure on classification in the field of NLP, and the viability of the method for using on different models and datasets.

2 Related Works

2.1 Adaptive Inference

Several pre-trained models have shown promising results in the field of NLP. For instance, BERT [1] and XLNet [18] have learned general knowledge from massive amounts of unlabeled data and applied it successfully to classification tasks in the field of NLP. However, these models have high computational costs and slow inference speeds. Many methods have been proposed to address these issues, including adaptive token deletion to speed up model inference. For example, TR-BERT [19] proposes an adaptive token reduction method that flexibly adjusts the number of layers of each token in inference to avoid redundant calculations. AdapLeR [20] dynamically eliminates the tokens that contribute less through the layer, thus shortening the length and increasing the calculation speed. GhostBERT [21] generates more features from the remaining features with very cheap operations. Thus the model has similar memory and computational cost to the pruning model. Other researchers regulate the number of coding layers within the adaptive regulatory model. For instance, FastBERT [10] exits simple samples from the model early based on the dynamics of the sample complexity to accelerate model inference. These adaptive methods have contributed to reducing computation and accelerating inference time.

However, they only consider simple tokens or deletion of samples, leading to a reduction in model accuracy. Complex samples are present in the samples without additional processing, resulting in a significant loss in model accuracy. In this paper, we propose adaptive training and inference models that identify complex samples to train a module separately and further subdivide the samples for adaptive inference, improving the model’s accuracy without affecting its inference speed.

2.2 Attention on Attention

Attention structures have achieved significant success in various models. However, the traditional attention mechanism cannot effectively obtain the correlation between attention results and query results. To overcome this deficiency, Jasdeep Singh et al. [15] proposed the AOA structure and applied it to the visual quiz task. AOA refines the attention mechanism to focus on the correlation between $Q$ and $V$. Lun Huang et al. [16] applied AOA to the image captioning model and achieved excellent results. To the best of our knowledge,
there has been no work in applying the AOA structure to the task of classification in the field of NLP to obtain the association between attention results and query results.

### 2.3 CNN Structure in Natural Language Processing

While Transformer structures have achieved good results in NLP, they mainly focus on the global information of text and lack the structure to obtain local information for semantic inference. In contrast, Yoon Kim et al.\[22\] achieved excellent results on multiple benchmarks using a simple CNN with few hyper-parameter adjustments and static vectors. However, the model only considers the local features of the text but ignores the global features. Zihang Jiang et al.\[23\] proposed a new span-based dynamic convolution, instead of self-focused heads, directly model local dependence. The new regular heads, together with the other self-attention heads, form a hybrid attention block that is more effective in both global and local context learning. Zhiliang Peng et al.\[24\] take advantage of convolution operation and self-attention mechanism to enhance representation learning. The Conformer model employs a concurrent structure to preserve both local features and global representations to the greatest extent. However, these models are simply splicing the global and local information, and this operation is difficult to obtain the connection between the two characteristics. In this paper, we use the local features and global features obtained by lightweight CNN for correlation fusion to obtain the enhanced features of complex samples for further inference. Our experimental results demonstrate that models with increased local information enhancement have a positive effect on improving model accuracy.

### 3 Methodology

![Model overall framework diagram](image)

**Fig. 2** Model overall framework diagram: The right figure (a) shows the flowchart of the training phase, and the left figure(b) shows the flowchart of the inference phase.
3.1 Method Overview

In the above model, during training (a), we first input all samples into the backbone model. All samples participate in the update of the backbone network parameters. We perform sample complexity analysis and screening on the output results, and save the status of the complex samples. When the number of saved states equals the batch size, we input them to the "sample feature enhancement module" to update its parameters. The final loss is the sum of the two modules. During inference (b), we feed the samples into the model. After passing through six layers of Transformer blocks, we set a simple sample exit mechanism for each layer to calculate the confidence of the obtained label score. If the confidence does not exceed the threshold $T_1$ (a hyperparameter), the exit mechanism is triggered, and the sample directly reaches the output layer. If the confidence does not trigger the exit mechanism, the sample passes through the entire model. Once the sample completes the inference of the backbone model, we perform complex sample analysis. If the sample complexity is above the threshold $T_2$ (a hyperparameter), it reaches the output layer. If the sample complexity is below the threshold $T_2$, it enters the "sample feature enhancement module" to extract local and global features, which are finally input to our AOA classifier for sample classification.

3.2 Model Training

Most of the existing models that speed up inference set thresholds during inference so that simple samples exit early\cite{25}, but they do not perform any additional processing on complex samples, thus losing model accuracy. Our approach is to train a complex sample feature enhancement module using the complex samples from the model during the training phase, and then utilize it in the subsequent inference phase. This approach addresses the issue of complex samples not undergoing extra processing, further improving the adaptability of the samples.

Given an input sentence $s = [W_0, W_1, ..., W_n]$, we pass it through an embedding layer to obtain a sequence of $e$, as shown in equation (1):

$$e = \text{Embedding}(s)$$  \hspace{1cm} (1)

This sequence is then passed through the encoder’s Transformer block, which performs layer-by-layer feature extraction, as shown in equation (2):

$$h_i = \text{Transformer}_i(h_{i-1})$$  \hspace{1cm} (2)

The output feature of the layer $i$ is represented by $h_i(i = -1, 0, 1, ..., L)$, where $h_{-1}$ is $e$ and $L$ is the number of Transformer layers. We refer to the output feature of the last layer as $H$. $H$ is passed through a classifier and a Softmax activation function. However, in conventional attention\cite{26}, when Query and Key/Value are not related\cite{16}, a set of normalized weights is still output for Query, which can lead to misleading information. To better extract
information from complex samples and improve the correlation between each state, we apply the AOA module to classification in the field of NLP for the first time. As shown in equation (3), (4), and (5):

$$I = \text{Self Attention}(H)$$

$$\text{AOA-Classifiers}(H) = \text{Mul}(\sigma(W_q^1H_q + W_v^1I + b^1), W_q^2Q + W_v^2I + b^2)$$

$\text{Mul}$ is the element multiplication operation, $\sigma$ is the Sigmoid operation, $W_q^1, W_v^1, W_q^2, W_v^2 \in \mathbb{R}^{D \times D}$, $D$ is the dimension of $H$.

$$P = \text{Softmax}(\text{AOA-Classifiers}(H))$$

The label score we obtain is denoted by $P$ (batch-size, label-number). By inputting $P$ into the Complex Sample Extraction Module, we can identify the complex samples that meet the threshold requirements in this batch-size (specified in Section 3.2.1). Meanwhile, the final state of all samples will participate in a cross-entropy loss to update the loss of the backbone network. The calculation method is shown in equation (6):

$$\text{loss}_1 = -\sum_{i=1}^{n} p_i \log p_i$$

$p_i$ in the above equation is the logits score of each label.

### 3.2.1 Complex Sample Extraction Module

Finding complex samples from samples is a prerequisite for further processing of complex samples. We propose a complex sample extraction module to address this problem. During training, we use the CSE-Train module to maintain concurrency. To prevent duplicate samples during inference, we use the CSE-Eval module. In contrast to Weijie Liu et al.’s[10] method of adding a student classifier to each layer during inference, which limits batch-size to one, our method can handle different batch-sizes during training without increasing training time. We accomplish this by comparing the complexity of the semantic analysis difficulty of different samples by using the two-norm of the probability distribution $P$, and filtering out simple samples through the $\text{Relu}$ activation function while finding the index of complex samples, as shown in equation (7):

$$\text{index} = Fi(\text{ReLU}(-P_2 + \alpha))$$

In the above equation, $Fi(\ast)$ functions to return the index of complex samples, and $\alpha$ is a learnable parameter.

After getting the index of the samples, we can easily find the complex samples. As shown in equation (8):

$$\text{hidden}_{diff} = \text{Cat}(\text{hidden}_{diff}, H[\text{index}])$$
hidden_{diff} extracts features at $H$ based on the complex sample index, and $Cat$ is a connect operation. After obtaining the hidden features of the complex sample, we input it into the Complex Sample Feature Enhancement module to further extract and train its features. It is worth noting that we will not feed complex samples directly into the next module, as it may not meet the batch size requirements. Therefore, we briefly store the characteristics of complex samples to match the size of the batch.

### 3.2.2 Characteristic Enhancement Module

Transformer structure is able to learn global features of text\cite{26, 27}, but it lacks the ability to capture local information. In the field of computer vision, the CNN structure has shown promising results due to its ability to fuse local characteristics of images\cite{28}. Inspired by this success, the NLP community has gradually introduced CNN structures. For instance, in short text analysis tasks, where sentence lengths are limited and the meaning can be expressed independently, CNN has been used for sentence classification\cite{22}. In this paper, we utilize the global information of text and introduce the Lightweight Convolutional Neural Network (LCNN) structure\cite{29} to capture local information. The range of the local information can be adjusted by tuning the number and width of convolution kernels. Finally, we fuse the global and local information using the AOA structure to obtain the final output. The specific structure is shown in the figure below:

![Diagram](image)

**Fig. 3** The left figure shows the structure diagram of the complex sample feature enhancement module, and the right figure shows the AOA structure diagram. First $H_{diff}$ and $H_{conv}$ generate state I through internal attention structure, then make a connection with $H_{conv}$, finally fuse the two attention points through element multiplication.

After obtaining the hidden state of the complex samples, we need to perform feature enhancement on the samples and continue training. The complex sample features then pass through several layers of Transformer, and the number of layers passed is related to the complexity of the samples, as shown in
Sample-adaptive Classification Inference Network

equation (9):

\[
\text{hidden}_{diff} = \text{Transformer}_i(\text{hidden}_{diff}) \quad i \in (1, 2...\beta) \tag{9}
\]

\(\text{hidden}_{diff}\) is a characteristic of the complex samples, \(\beta\) is dynamically variable and obtained from model training. To address the disadvantage of the lack of local information in self-attention, we input the resulting complex sample features into a lightweight CNN to obtain local sample features, as shown in equation (10):

\[
\text{hidden}_{cnn} = \text{Linear}(\text{LConv}(\text{Linear}(\text{hidden}_{diff}))) \tag{10}
\]

\(\text{LConv}\) is a lightweight convolution operation, \(\text{hidden}_{cnn}\) are the resulting sample local characteristics.

Finally, the resulting two sets of eigenvectors are fused proportionally through the AOA module to obtain the final output. As shown in equation (11):

\[
\text{hidden}_{\text{final}} = \text{AOA}((\text{hidden}_{diff}, \text{hidden}_{diff}, \varepsilon \ast \text{hidden}_{cnn}) \tag{11}
\]

Where the \(\text{hidden}_{\text{final}}\) is the final output of complex samples, and \(\varepsilon\) is the proportion of global and local features fused, obtained from learning.

After obtaining the final output of the complex samples, it is fed into the classification layer for classification and to calculate the loss of the complex samples. The complex sample features enhancement module is then updated with the loss of the complex samples, as shown in equation (12) and (13):

\[
\mathcal{P}_{diff} = \text{Softmax}((\text{AOA}_{\text{Classifiers}}(H_{\text{diff}}))) \tag{12}
\]

\[
\text{loss}_2 = -\sum_{i=1}^{N} p^i_{diff}\log p^i_{diff} \tag{13}
\]

Among \(p^i_{diff}\) is the logits score of each label, and \(N\) is the number of labels.

### 3.2.3 Loss Calculation

As mentioned above, our model produces two types of losses. The first is the loss of all samples through the backbone model \((\text{loss}_1)\), while the second is the loss of the complex sample feature enhancement module \((\text{loss}_2)\) during training. The ultimate loss is the sum of these two losses. It is important to note that both losses are independent of updating the parameters of different modules. \(\text{loss}_1\) is used to update the parameters in the backbone model, while \(\text{loss}_2\) is used to update the complex sample feature enhancement module.

\[
\text{loss} = \text{loss}_1 + \text{loss}_2 \tag{14}
\]
3.3 Model Inference

Currently, pre-trained models are rapidly developing in the direction of improving inference speed, making their application prospects even broader and more cost-effective. These methods typically involve either a sentence-level exit mechanism or token deletion at the word level\cite{30}\cite{20}. While these methods can significantly improve model inference speed, they also lead to a certain degree of accuracy loss. The main problem with these methods is that they focus on withdrawing or deleting simple samples while ignoring the fact that there are not only simple samples, but also complex samples. It is often these complex samples that lead to a loss of model inference accuracy. Our trained complex sample feature enhancement module can be used to address these complex samples by maximizing sample adaptation, thereby addressing the limitations of existing models. In the inference stage, we take into account the differences among samples and apply corresponding processing methods. During model training, we divide samples into common samples and complex samples. In the inference stage, we further subdivide samples into simple samples that can be withdrawn in advance, normal samples that go through the entire backbone model, and complex samples that require further judgment. By dynamically inferring different types of samples, we can improve the accuracy of the inference. Additionally, due to the early exit of simple samples, our model can accelerate the inference speed of the model to some extent. The process and training stage of inference are basically the same, but the difference lies in the addition of an adaptive withdrawal mechanism for simple samples and a complex sample extraction mechanism for the inference stage. The following two sections will detail these mechanisms.

3.3.1 Sample Adaptive Exit Mechanism (SSC)

In the inference stage, samples have simple labels for their exact inference even if they do not pass through the entire model. To speed up the inference process without compromising model accuracy, we implement a simple sample exit mechanism during inference. If a sample meets the exit condition, it can exit in time, ensuring that the model can accelerate its inference speed to some extent. To implement this mechanism, we first pass the sample through the first five layers of the Transformer block. Starting from the sixth layer, we begin calculating the credibility of each label score for the sample.

\[
P_i = \text{Softmax}(\text{AOA-Classifiers}_i(H_i)) \quad (5 < i <= 12) \tag{15}
\]

\(H_i\) represents the characteristics when the sample passes through layer \(i\)th Transformer, \(\text{AOA-Classifiers}_i\) representing the layer \(i\)th classifier. In order to further analyze the complexity of samples, the accuracy of the model is improved to the maximum extent. We set a hyperparameter \(\lambda\) to adjust the number of layers participating in the calculation when the confidence of the label score is less than the set threshold \(T_1\). If the confidence is below \(T_1\), the sample exits the model directly; otherwise, it continues to input into the next
Sample-adaptive Classification Inference Network

Transformer. The calculation of $t$ is shown in equation (16):

$$ t = \frac{\sum_{k>6}^{k+\lambda} \sum_{i=1}^{N} p_k^i \log p_k^i}{-\log N} $$

$p_i$ is the probability of each category, $N$ is the total number of categories, $K$ represents the number of layers of Transformer, and $\lambda (1 < \lambda < 5)$ is used to control the number of layers through the exit mechanism, which is set to 2 in this paper. The above equation indicates that we will continuously count the values of the $\lambda$ layer to determine whether the sample is a simple sample.

### 3.3.2 Complex Sample Extraction and Inference Mechanism

During the inference stage, we also need to identify complex samples, which are extracted differently from the training stage. Since we set the batch-size to one in the inference stage to fully adapt the sample through the model, we assess the sample’s complexity after it passes through the backbone model. At this stage, the sample does not exit the model prematurely after passing through the 12-layer Transformer of the backbone model.

$$ P = \text{Softmax}(\text{AOA}_\text{Classifiers}_{12}(H_{12})) $$

(17)

After the resulting probability distribution $P$, we need to further analyze the complexity of the sample, as shown in equation (18):

$$ \text{score} = \frac{1}{\log N \prod_{i=1}^{N} (p_i + C)} $$

(18)

$p_i$ is the probability of each category; $N$ is the total number of label categories; and $C$ is the hyperparameter (generally set to 0.1). The resulting score is the complexity score for this sample. Then score greater than the complexity threshold T2. The sample is input into the already trained feature enhancement module. Samples that do not reach complex requirements will be directly output directly.

Finally, all samples are adapted through several layers of Transformer. From this, we can determine that samples in the model pass through a minimum of 7 layers of Transformer blocks and a maximum of $12+\beta$ layers of Transformer blocks. This adaptation is based solely on the sample itself.

### 4 Experimental Results

In this section, we verify the effectiveness of our model on six classification tasks datasets in the field of NLP (three English and three Chinese). We also conduct ablation experiments on different modules to explain the experimental results.
4.1 Model and Datasets

Our method can be applied to multiple models with promising results. We applied the sample adaptive training and inference mechanism to three commonly used baseline models and compared them with the original baseline.

BERT is the NLP pre-trained model proposed by Google and essentially constructing a multilayer bidirectional Encoder network using the Transformer structure[1]. Where BERT-Base uses 12-layer Transformer blocks as the main framework of the algorithm. The number of parameters is about 110M.

FastBert is a new inference speed improvement method proposed by Peking University and other institutions[10]. Compared with pure student distillation, it has higher certainty and can balance the effect and speed by itself.

TernaryBERT is a model proposed by Huawei that adopts knowledge distillation and fine-tuning BERT weights to reduce model parameters and accelerate inference[4].

We will compare our method on three English datasets (SST-2, Newsgroups, Yelp-full) and three Chinese datasets (Book review, WeiBo, Lcqmc). The task of SST-2 is to analyze the sentiment of a given sentence, with 67,350 training samples and 1,821 test samples; the task of Newsgroups is to classify a given text, with 20,000 training samples and 7,600 test samples; the task of Yelp-full is to analyze the emotion of a given sentence, with 130,000 training samples and 10,000 test samples; the task of WeiBo is to analyze the sentiment of a given sentence, with 10,000 training samples and 2,000 test samples; the task of Book review is to analyze the sentiment of a given sentence, with 10,000 training samples and 1,600 test samples; the task of Toutiaonews is to analyze the sentiment of a given sentence, with 230,000 training samples and 76,000 test samples.

<table>
<thead>
<tr>
<th>Model/Datasets</th>
<th>SST-2</th>
<th>NewsGroups</th>
<th>Yelp-full</th>
<th>WeiBo</th>
<th>Toutiaonews</th>
<th>Book review</th>
</tr>
</thead>
<tbody>
<tr>
<td>BERT</td>
<td>92.5</td>
<td>86.5</td>
<td>65.9</td>
<td>97.4</td>
<td>73.0</td>
<td>86.7</td>
</tr>
<tr>
<td>FastBERT</td>
<td>91.8</td>
<td>85.2</td>
<td>64.6</td>
<td>97.2</td>
<td>70.8</td>
<td>86.6</td>
</tr>
<tr>
<td>TernaryBERT</td>
<td>92.1</td>
<td>86.1</td>
<td>65.7</td>
<td>96.9</td>
<td>71.8</td>
<td>86.7</td>
</tr>
<tr>
<td>BERT+Ours</td>
<td>93.1</td>
<td>87.1</td>
<td>66.3</td>
<td>97.8</td>
<td>73.2</td>
<td>87.6</td>
</tr>
<tr>
<td>FastBERT+Ours</td>
<td>92.6</td>
<td>86.3</td>
<td>65.2</td>
<td>97.3</td>
<td>71.6</td>
<td>87.2</td>
</tr>
<tr>
<td>TernaryBERT+Ours</td>
<td>92.5</td>
<td>86.2</td>
<td>66.0</td>
<td>97.1</td>
<td>72.3</td>
<td>86.9</td>
</tr>
</tbody>
</table>

In the above table, we controlled the proportion of simple samples between 20% and 25% by adjusting the hyperparameter $T_1$. We controlled the proportion of complex samples between 5% and 10% by adjusting the hyperparameter $T_2$. All experiments were conducted under this proportion, except for Section 4.4. The experimental results show that our method can be applied to different models and achieves certain effects in three English and three Chinese datasets. This fully demonstrates the feasibility of our method.
4.2 Ablation Experiments

We used the text adaptive training and inference mechanism, the text enhancement structure (obtaining local features without LCNN), and the AOA structure as the final classifier. The effects of these mechanisms and modules are demonstrated by ablation experiments. Below we will show it with BERT as the baseline.

Table 2  Data from the ablation experiments

<table>
<thead>
<tr>
<th>Model/Datasets</th>
<th>SST-2</th>
<th>NewsGroups</th>
<th>Yelp-full</th>
<th>WeiBo</th>
<th>Toutiaonews</th>
<th>Book review</th>
</tr>
</thead>
<tbody>
<tr>
<td>BERT</td>
<td>92.5</td>
<td>86.5</td>
<td>65.9</td>
<td>97.4</td>
<td>73.0</td>
<td>86.7</td>
</tr>
<tr>
<td>NO AUTO</td>
<td>92.7</td>
<td>86.6</td>
<td>66.0</td>
<td>97.5</td>
<td>73.0</td>
<td>87.2</td>
</tr>
<tr>
<td>NO AOA</td>
<td>93.0</td>
<td>86.8</td>
<td>66.2</td>
<td>97.7</td>
<td>73.3</td>
<td>87.4</td>
</tr>
<tr>
<td>NO LCNN</td>
<td>93.0</td>
<td>87.2</td>
<td>66.0</td>
<td>97.6</td>
<td>73.1</td>
<td>87.3</td>
</tr>
<tr>
<td>ALL</td>
<td>93.1</td>
<td>87.1</td>
<td>66.3</td>
<td>97.8</td>
<td>73.2</td>
<td>87.6</td>
</tr>
</tbody>
</table>

As shown in the table above, BERT is on the BERT-base model, No AUTO is removed on the basis of our model to the text adaptive training and inference mechanism, NO LCNN is removed by the text enhancement module using LCNN in the model, NO AOA is the model classifier with ordinary attention, ALL is the mechanism and design module all applied to the model. As shown in the figure above, our text adaptive training and inference mechanism, as well as the AOA structure, have shown positive effects on all types of datasets. However, we found that the LCNN structure has a subtle effect on both Chinese and English datasets, and even exhibits negative effects in some English datasets. Our analysis suggests that this may be due to the difference between Chinese and English sentences. English sentences are composed
of words with complete meanings, whereas Chinese sentences are constructed from individual words to form complete sentences. However, the LCNN-based text enhancement module mainly enhances the local representation of text, and Chinese text may be more sensitive to local information. Therefore, the text enhancement module based on LCNN plays a greater role in Chinese text.

4.3 Calculation and Quantity Analysis

Table 3 The accuracy of the model on each datasets

<table>
<thead>
<tr>
<th>Model/Datasets</th>
<th>SST-2</th>
<th>NewsGroups</th>
<th>Yelp-full</th>
<th>WeiBo</th>
<th>Toutiaonews</th>
<th>Book review</th>
</tr>
</thead>
<tbody>
<tr>
<td>BERT</td>
<td>21862M</td>
<td>21862M</td>
<td>21862M</td>
<td>21862M</td>
<td>21862M</td>
<td>21862M</td>
</tr>
<tr>
<td>FastBERT</td>
<td>6056M</td>
<td>5682M</td>
<td>11266M</td>
<td>5412M</td>
<td>9792M</td>
<td>4906M</td>
</tr>
<tr>
<td>TernaryBERT</td>
<td>5441M</td>
<td>5236M</td>
<td>8021M</td>
<td>5823M</td>
<td>7569M</td>
<td>5029M</td>
</tr>
<tr>
<td>BERT+Ours</td>
<td>19862M</td>
<td>21277M</td>
<td>20532M</td>
<td>21029M</td>
<td>20263M</td>
<td>20167M</td>
</tr>
<tr>
<td>FastBERT+Ours</td>
<td>7236M</td>
<td>5598M</td>
<td>10235M</td>
<td>6894M</td>
<td>10894M</td>
<td>5178M</td>
</tr>
<tr>
<td>TernaryBERT+Ours</td>
<td>5025M</td>
<td>4938M</td>
<td>7768M</td>
<td>5412M</td>
<td>7038M</td>
<td>5125M</td>
</tr>
</tbody>
</table>

Based on the above table, we can conclude that our method adds little to the computational cost to obtain better results, and in some datasets, the computational cost may even decrease. Our analysis suggests that this is due to the varying difficulty of the samples collected in each datasets. Additionally, the Transformer module occupies the vast majority of the computational cost in the model. If only considered the computational cost produced through the Transformer block, in our experiments, we found that on average, simple samples pass through eight Transformer blocks, while medium and complex samples pass through an average of twelve and fifteen Transformer blocks, respectively. By comparing with the baseline model, we obtained the following calculation equation:

\[
\text{speed} = \frac{\text{simple} \times 8 + \text{medium} \times 12 + \text{difficult} \times 15}{12} \tag{19}
\]

Suppose 70% simple, 20% moderate, and 10% complex samples in datasets A. The acceleration ratio was calculated as 1.26.

Suppose 30% simple, 50% moderate, and 20% complex samples in datasets B. The acceleration ratio was calculated at 1.05.

Suppose 10% simple, 40% moderate, and 50% complex samples in datasets C. The acceleration ratio was calculated as 0.916.

We can find that the acceleration ratio of the model is correlated with the difficulty of the sample average; this is actually reasonable and the result of the adaptation of our model.

4.4 Adaptive Inference

The degree of adaptation of our model is closely related to the parameter setting. The model can choose the appropriate hyperparameters according to
different requirements and application scenarios. Specifically, we can control the inference speed of the model by regulating $T_1$ and $T_2$ by regulating $T_1$. We performed an adaptive inference experimental analysis on the SST-2 dataset on the BERT-base model. The experimental results were compared by adjusting $T_1$ and $T_2$.

From the above figure, we can find that when $T_1$ is larger, the proportion of samples withdrawn from the model exit increases; when $T_2$ is smaller, the proportion of samples with features enhancement increases. By adjusting the proportion of sample dropout and feature enhancement, we can modulate the inference speed and accuracy of the model. We will continue to count the accuracy rate and sample average calculation under different hyperparameters:

**Fig. 5** Distribution of sample proportions under different hyperparameters

**Fig. 6** Accuracy and average FLOPs for different hyperparameters
In the figure above, we can find that the model performs best in the case of $T_1=0.9$ and $T_2=14$, because if the sample exits early with too many samples, although it can effectively speed up the inference, it also affects the accuracy of the model to a large extent. The number of sample samples of feature enhancement should be basically consistent with the level of training samples, so that the role of the sample enhancement module can be maximized.

5 Conclusion

This paper presents a sample-adaptive training and inference model. This method realizes the adaptive inference of complex samples in the inference stage by training the complex sample feature enhancement modules, which compensates for the defect that the existing models do not differentiate the complex samples. In order to realize the complex sample feature enhancement, this paper extracts the sample local features through the convolution module, and the Transformer module extracts the global features of the sample. Later, the two features are fused through the AOA structure to obtain enhanced attention with both local and global features. In inference, we set the simple sample exit mechanism, which will satisfy the exit mechanism, in order to speed up the inference of the model. For those complex samples, our feature enhancement module will reason after feature enhancement, so as to improve the accuracy of the model. The adaptive mechanism is more perfect and reasonable than other models, and the method proposed in this paper achieves good results under both objective indicators and subjective performance.

Declarations

- We express our sincere thanks to the anonymous reviewers for their helpful comments and suggestions to raise the standard of this paper.
- This work was supported partially by the National Key R&D Program of China (grant number 2020YFC1512601) and National Natural Science Foundation of China (grant number 62106064).
- The content of the manuscript does not involve any ethical issues.
- All authors contributed to the study conception and design. Material preparation, data collection and analysis were performed by Juan Yang, Guanghong Zhou, Ronggui Wang and Lixia Xue. The first draft of the manuscript was written by Guanghong Zhou and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

References

Sample-adaptive Classification Inference Network


Sample-adaptive Classification Inference Network


