Adaboost with Auto-Evaluation for Conversational Models

Juncen Li¹, Ping Luo²,³, Ganbin Zhou²,³, Fen Lin¹, Cheng Niu¹

¹ WeChat Search Application Department, Tencent, China
² Key Lab of Intelligent Information Processing of Chinese Academy of Sciences (CAS), Institute of Computing Technology, CAS, Beijing 100190, China
³ University of Chinese Academy of Sciences, Beijing 100049, China

{juncenli,felicialin,niucheng}@tencent.com, luop@ict.ac.cn

Abstract

We propose a boosting method for conversational models to generate more human-like dialogs. In our method, we consider the existing conversational models as weak generators and apply the Adaboost to update those models. However, conventional Adaboost cannot be directly applied on conversational models, since conventional Adaboost cannot adaptively adjust the weight on the instance for subsequent learning. This results from the conventional methods based on the simple comparison between the true output y (to an input x) and its corresponding predicted output y', cannot effectively evaluate the learning performance on x. To address this issue, we develop the Adaboost with Auto-Evaluation (called AwE). In AwE, an auto-evaluator is proposed to evaluate the predicted results, which makes Adaboost applicable to conversational models. Furthermore, we present the theoretical analysis that the training error drops exponentially fast only if certain assumption over the proposed auto-evaluator holds. Finally, we empirically show that AwE visibly boosts the performance of existing single conversational models and also outperforms the other ensemble methods for conversational models.

1 Introduction

In the light of these advances, recent years witness an increasing research interests in applying the end-to-end neural methods to conversational models [Shang et al., 2015; Sordoni et al., 2015; Serban et al., 2016; 2017]. These previous studies aim to improve conversational models. First, more sophisticated neural architectures are proposed, such as attention mechanism [Bahdanau et al., 2015] and hierarchical neural network [Serban et al., 2016], are proposed to depict the mapping between input and output more carefully. Second, some other learning methods are applied to updated conversational models. For example, reinforcement learning [Li et al., 2016], adversarial learning [Li et al., 2017] and adaptation methods [Li et al., 2018] are used to modify conversational models. However, two problems of conversational models are still not solved completely. Firstly, existing conversational models may generate responses which are not suitable to input posts in any scenarios. Secondly, these models tend to generate several highly generic responses, for example, “I don’t know”.

With the above two problems, the performance of existing conversational models are unsatisfactory. Here, they can be seen as weak generators. To update weak generators, we extend the ensemble method Adaboost for conversational models. Adaboost [Freund and Schapire, 1997] is an effective ensemble framework to improve the performance of supervised learning (originally for single-variable output). The key to Adaboost is its adaptive feature where subsequent weak learners are tweaked in favor of those instances on which previous weak-performance learners. Thus, it needs an automatic method to evaluate the learning performance on each instance.

However, for conversational models the automatic evaluation is not trivial. Consider the dialog generation task. Let y and y' be the true and predicted response to an input post x. To be a good response to x, y' is not required to be exactly equal to y, but is only required to be grammatically correct and semantically relevant to input x. Thus, for conversational models, the evaluation of the predicted responses is not trivial, therefore conventional Adaboost cannot be applied on conversational models directly.

In this paper, we study the extension of Adaboost for conversational models. To this end, we develop the framework of Adaboost with Auto-Evaluation (called AwE). In AwE, an auto-evaluator is proposed to evaluate predicted results. Specifically, motivated by the discriminator in Generative Adversarial Nets [Goodfellow et al., 2014], a classifier is trained to distinguish the true and predicted responses for the input posts. Then, we believe that the conversational model performs well on x only if the classifier makes a false-positive error on x. Thus, this classifier can be used as an auto-evaluator to adjust the weights on instance for the next-round learning in Adaboost. Furthermore, we theoretically analyze the training error of AwE. The training error drops exponentially fast only if certain assumption over the proposed auto-evaluator holds. Finally, we do some empirical experiments to evaluate our method. We demonstrate that AwE visibly
boosts the performance of single model and also outperforms the other ensemble methods for conversational models.

2 Related Work

In the light of end-to-end neural system of statistical machine translation (SMT) [Yin et al., 2016; Cho et al., 2014; Bahdanau et al., 2015], researches on neural conversational models have made some progress. Shang et al. [2015] introduced three types of encoding schemes as extensions of attention method. They found that hybrid scheme performs better than the other two schemes in generating responses. Instead of focusing on one-round dialog, Serban et al. [2016] devised a hierarchical neural network. They encoded a sequence of words into an utterance vector and used the utterance vector of previous sentences as context information. Then they [2017] extended the hierarchical neural network by adding a parallel RNN encoder, which encodes the high-level coarse tokens, into the previous framework. Another way to improve conversational models is to solve the high frequency responses problem and increase the response diversity. Jiwei et al. [2016] brought reinforcement learning into the dialog system to overcome the high frequency responses challenge.

AdaBoost was proposed by Freund and Schapire [1997]. It was applied to improve performance of word recognition in handwritten documents [Schwenk and Bengio, 1998], face recognition [Sun et al., 2012]. It was also used to update structured learning with explicit evaluation labels [Cortes et al., 2014]. All of those tasks can be automatically evaluated by simple comparison between the true output $y$ (to an input $x$) and its corresponding predicted output $y'$. However, to our best knowledge, AdaBoost has not been employed on conversational models.

We propose AwE which employs Adaboost to update conversational models. Our method designs an auto-evaluator inspired by the discriminator of Generative Adversarial Nets (GAN) [Goodfellow et al., 2014]. GAN and its variants, such as LAPGAN [Denton et al., 2015] and DCGAN [Radford et al., 2015], have made great progress on image generation. GAN consists of two components: a generator and a discriminator. The generator generates images which are similar to real images. And the discriminator differentiates real images and fake images which are generated by the generator. For application of GAN on natural language processing (NLP), Lantao et al. [2017] proposed SeqGAN to generate text.

3 Adaboost with Auto-Evaluation

In this section, we introduce AdaBoost with Auto-Evaluation (AwE). Firstly, we describe the training process of AwE. As shown in Fig.1, the main training process of AwE is to train several models iteratively. We regard every training epoch as an epoch module and the framework of our method is to combine those modules. Then we propose a Weighted Beam Search method as the inference method for AwE. Finally, we prove an upper bound of the training error when certain assumption on the auto-evaluators holds.

3.1 Training Process

The epoch module is shown in Fig.1(a). It has two inputs. The first input is a dataset $D = \{x_i, y_i\}_{i=1}^N$, where $x_i$ is input post, $y_i$ is output response, $N$ is the number of input-output pairs in the dataset. The other input is a weight set $w^{(k-1)}$, $w^{(k-1)}$ is leveraged to control the attentions of convolutional model (CM) on input-output pairs. A CM mainly focuses on the input-output pairs where the CM of previous epoch module obtains unsatisfactory performance. In addition, we initialize all weights equally, which means every pair is equally important at the beginning.

There are two main models in the epoch module: conversational model (CM) and auto-evaluation (AE). Our method can be easily used on the neural conversational models which are based on sequence-to-sequence format, such as [Cho et al., 2014; Sutskever et al., 2014; Bahdanau et al., 2015; Serban et al., 2016; Shang et al., 2015]. Here, the method in [Cho et al., 2014] is adopted. We can utilize $w^{(k-1)}$ and $D$ to train CM. Using the trained model, we then generate
Algorithm 1 Auto-Evaluation AdaBoost

Require:
Dataset, D = (x_i; y_i)_{i=1}^N, where x_i is input, y_i is output, N is the number of input-output pairs
Conversational model, CM
Auto-evaluator, AE
Iteration number, K
Ensure: Ensemble model g_e(x, y)

1: w^{(0)} ← (\frac{1}{N}, \ldots, \frac{1}{N}) // w is the set of weights
2: for k = 1 to K do
3: g^{(k)} ← CM \left( D, w^{(k-1)} \right)
4: e^{(k)} ← AE \left( \left\{ D, D_g^{(k)} \right\} \right)
// D_g^{(k)} is the generated dataset predicted by the trained CM
5: f^{(k)}(x_i) ← \begin{cases} 1, & e^{(k)}(x_i, y_i') = 1 \\ -1, & e^{(k)}(x_i, y_i') = -1 \end{cases}, \forall i
6: w^{(k)} ← \frac{1}{\text{Z}} \prod_{i=1}^{N} w^{(k-1)} \prod_{i=1}^{N} I \left[ f^{(k)}(x_i) \neq 1 \right]
7: \alpha^{(k)} ← \frac{1}{2} \log \left( \frac{1-e^{(k)}}{e^{(k)}} \right)
8: \forall i, w^{(k)}(x_i, y_i') ← \frac{1}{w^{(k-1)}} \exp \left( -\alpha^{(k)} f^{(k)}(x_i) \right)
// Z is the sum of w^{(k)} that are used for normalization
9: end for
10: return g_e(x, y) = \sum_{k=1}^{K} \alpha^{(k)} g^{(k)}(x, y)

Figure 1. (b). In the whole process, we combine several epoch modules. w^{(k)} can be regarded as the connector between two epoch modules. w^{(k)} is output by the k-th epoch module and is the input of the (k+1)-th epoch module. It stores the information of which pairs are unsatisfactorily learned and guides the training of CM in the (k+1)-th epoch module.

3.2 Weighted Beam Search for Sequence Generation

In the inference process, we need to generate an output y' = \{y^{(1)}, y^{(2)}, \ldots, y^{(L)}\} (where L is the length of the output) for a given input x. Specifically, we select the y' by maximizing probability with g_e:

\[ y' = \arg \max_y g_e(x, y) \]
\[ = \arg \max_y \sum_{k=1}^{K} \alpha^{(k)} g^{(k)}(x, y) \]
\[ = \arg \max_y \sum_{k=1}^{K} \alpha^{(k)} p^{(k)}(y|x) \]

Using the above equation, we must traverse all plausible outputs. This method is time-consuming. To reduce the time-complexity, we propose the Weighted Beam Search method which generates the output word by word. Specially, when generating the first words, we obtain the conditional probability of first words using:

\[ p(y^{(1)}|x) = \sum_{k=1}^{K} \alpha^{(k)} p^{(k)}(y^{(1)}|x) \]

The first words of outputs can be selected according to p(y^{(1)}|x). M (beam search size) words with highest p(y^{(1)}|x) are selected as possible words. In the step of generating sub-sequence \{y^{(1)}, \ldots, y^{(n)}\} (n > 1), Weighted Beam Search firstly uses the added conditional probability as the ensemble conditional probability:

\[ p(y^{(n)}|x, y^{(1)}, \ldots, y^{(n-1)}) = \sum_{k=1}^{K} \alpha^{(k)} p^{(k)}(y^{(n)}|x, y^{(1)}, \ldots, y^{(n-1)}) \]
probability of sub-sequence \( p(y^{(1)}, \ldots, y^{(n)} | x) \) using:
\[
p(y^{(1)}, \ldots, y^{(n)} | x) = p(y^{(n)} | x, y^{(1)}, \ldots, y^{(n-1)}) 
\cdot p(y^{(1)}, \ldots, y^{(n-1)} | x)
\]  (4)

Then we can select \( M \) possible sub-sequences \( \{y^{(1)}, \ldots, y^{(n)}\} \) with highest conditional probability \( p(y^{(1)}, \ldots, y^{(n)} | x) \).

### 3.3 Analyzing the Training Error of AwE

Here, we use \( y_i' \) to represent the output generated by the ensemble model and \( e(x_i, y_i') \) to represent the evaluation result of \( y_i' \) to \( x_i \). If \( e(x_i, y_i') = 1 \), then \( y_i' \) is suitable to \( x_i \) and vice-versa. Thus, \( \frac{1}{N} \sum_{i=1}^{N} I[e(x_i, y_i') \neq 1] \) can represent the training error of AwE. Next, we propose some assumptions on the auto-evaluator and then propose an upper bound to this training error based on this assumption.

**Assumption 1.**
\[
e(x_i, y_i') = \text{sgn} \left( \sum_{k=1}^{K} \alpha^{(k)} f^{(k)}(x_i) \right)
\]  (5)

The assumption actually requires the evaluation on \( y_i' \) (predicted by the ensemble method) to be the linear sum of the evaluation results from the \( K \) individual auto- evaluators. We argue that this assumption is reasonable since each individual auto-evaluator \( e^{(k)} \) contributes the final evaluation with the weight \( \alpha^{(k)} \). With this assumption we have the following theorem for the upper bound of training error for AwE.

**Theorem 1.** The upper error bound of AwE on training data is as follows:
\[
\frac{1}{N} \sum_{i=1}^{N} I[e(x_i, y_i') \neq 1] \leq \exp \left( -2 \sum_{k=1}^{K} \gamma_k^2 \right)
\]  (6)

where \( \gamma_k = \frac{1}{2} - e^{(k)} \).

**Proof.** First, let
\[
f(x_i) = \text{sgn} \left( \sum_{k=1}^{K} \alpha^{(k)} f^{(k)}(x_i) \right)
\]  (7)

Then, based on the theorem on the upper error bound for Adaboost [Freund and Schapire, 1997] we have
\[
\frac{1}{N} \sum_{i=1}^{N} I[f(x_i) \neq 1] \leq \exp \left( -2 \sum_{k=1}^{K} \gamma_k^2 \right)
\]  (8)

Then, to prove Theorem 1, we need to prove
\[
\frac{1}{N} \sum_{i=1}^{N} I[e(x_i, y_i') \neq 1] = \frac{1}{N} \sum_{i=1}^{N} I[f(x_i) \neq 1]
\]  (9)

The above Eq. 9 can be proved as follows:

---

### Table 1: Some statistics of the dataset

<table>
<thead>
<tr>
<th></th>
<th>posts</th>
<th>responses</th>
<th>pairs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training</td>
<td>536,639</td>
<td>542,083</td>
<td>773,315</td>
</tr>
<tr>
<td>Validation</td>
<td>20,000</td>
<td>25,086</td>
<td>28,949</td>
</tr>
<tr>
<td>Test</td>
<td>1000</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

From Theorem 1, we get the conclusion that with the growth of \( K \), the upper error bound of the training data decreases exponentially. In other words, we can increase \( K \) to improve the performance of the ensemble method over the training data. However, when \( K \) becomes too large, the model may overfit the training data. Thus, finding a suitable value of \( K \) is important. In this paper, we select the value of \( K \) according to performance of AwE on validation dataset.

### 4 Experiments

We examine the performance of our method on dialog generation tasks and compare with three benchmarks in experiments. Most of conversational models based on sequence-to-sequence format can be used in our method. For convenience, we choose RNN Encoder-Decoder with GRU [Cho et al., 2014] as the conversational model in our experiments. For fair comparison, all the benchmarks are based on the RNN Encoder-Decoder with GRU.

### 4.1 Dataset

We collect nearly 14 million post-response pairs from Tencent Weibo. Removing spams and advertisements from that dataset, there are only 803,716 high-quality post-response pairs retained. We randomly divide this data into training, validation and test set with no overlap posts. In other words, we insure that there are no posts appearing in two different sets. Table 1 shows some statistics of the dataset used in this paper. Then we build the dictionary with Chinese characters and high frequency Chinese words. The dictionary consists...
We use three benchmarks in this paper which are all based on RNN Encoder-Decoder with GRU.

- Seq2Seq: This method is RNN Encoder-Decoder with GRU [Cho et al., 2014].
- Maximum: It is an ensemble method which is similar to the ensemble method in [Zhou et al., 2017]. It combines the results of several generators by the way of selecting results with maximum scores.
- Bagging: This ensemble method is widely used in reading comprehension and question answering [Wang et al., 2017a; 2017b; Seo et al., 2017]. It is an ensemble method which combines several different models by averaging scores of those models and choosing the response with highest average score.

### 4.3 Experimental Details
The experimental details in this paper are as follows:

- We use 1-layer GRU with 512 cells for both the encoder and the decoder.
- We use different word embeddings for the encoder and the decoder as suggested in [Shang et al., 2015]. Both embedding dimensions are set to 128.

- We initialize all parameters with the uniform distribution between -0.1 and 0.1. And We set the minibatch size to 256.
- We use beam search method to do the generation and we set beam size to 10.

### 5 Results and Analysis
We use two evaluation methods to compare the performance between our method and three benchmarks: BLEU and human evaluation method [Shang et al., 2015].

#### 5.1 Human Evaluation
We use human evaluation method to evaluate our models and three benchmarks referring to [Shang et al., 2015]. To prevent human annotation bias, we mix generated results of all models up and let four labelers score the same result set independently. All the labelers come from a professional company and have at least one-year experience of labeling dialog system. The human evaluation examples are shown in Table 2. The score ranges from 0 to 2 indicating bad, normal and good respectively.

- Bad(0): The generated response is not semantically relevant to the post or there are some grammatical mistakes in the response.
- Normal(1): The generated response has no grammatical mistakes and is semantically relevant to the post. But it is a high frequency response or it can only be suitable to the post in some specific scenarios.
- Good(2): The generated response is free of mistakes and semantically relevant to the post. Additionally, it is a very satisfying response to the post.

#### 5.2 Results
Table 3 shows the BLEU (BLEU-4) of our methods and three benchmarks. We can find that AwE outperforms three benchmarks. Also we can see that there is nearly no difference among three benchmarks. The main reason we conjecture is that the two ensemble benchmarks do not generate more similar responses to the reference responses than single model. Instead, they generate more high frequency responses, which is shown in Table 4. High frequency responses are the responses that appear many times in the dataset and can respond to many different posts, such as “I don’t know”, “I...
<table>
<thead>
<tr>
<th>Methods</th>
<th>Mean Score</th>
<th>Good(2)</th>
<th>Normal(1)</th>
<th>Bad(0)</th>
<th>Agreement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Seq2Seq</td>
<td>0.903</td>
<td>14.1%</td>
<td>62.2%</td>
<td>23.7%</td>
<td>0.480</td>
</tr>
<tr>
<td>Maximum</td>
<td>0.905</td>
<td>11.7%</td>
<td>67.1%</td>
<td>21.2%</td>
<td>0.505</td>
</tr>
<tr>
<td>Bagging</td>
<td>0.906</td>
<td>12.1%</td>
<td>66.4%</td>
<td>21.5%</td>
<td>0.436</td>
</tr>
<tr>
<td>AwE</td>
<td><strong>0.989</strong></td>
<td><strong>15.2%</strong></td>
<td><strong>68.5%</strong></td>
<td><strong>16.3%</strong></td>
<td><strong>0.412</strong></td>
</tr>
</tbody>
</table>

Table 5: Human evaluation results

<table>
<thead>
<tr>
<th>No</th>
<th>Post</th>
<th>Single method</th>
<th>Ensemble method</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>去台湾有什么值得买</td>
<td>是当然值得的</td>
<td>Seq2Seq</td>
</tr>
<tr>
<td></td>
<td>What is worth buying in Taiwan</td>
<td>当然是值得值得的</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>你为什么要打我吗</td>
<td>我不明白你要说什么</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Why do you want to beat me?</td>
<td>I don't know what you are talking about</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>一加一等于几</td>
<td>一加一等于二</td>
<td>Seq2Seq</td>
</tr>
<tr>
<td></td>
<td>One plus one is two</td>
<td>I don't know what you are talking about</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>学校附近哪里有好</td>
<td>Chinese Shandong</td>
<td>Ensemble method</td>
</tr>
<tr>
<td></td>
<td>是哪里有好地方来学习</td>
<td>Luoyang is good</td>
<td></td>
</tr>
</tbody>
</table>

Table 6: Responses generated by several methods

... don’t know what you are talking about”, “Such a simple question”. Because nearly all top 100 responses of training dataset are high frequency responses, we use those responses to represent high frequency responses in Table 4. But, we must make it clear that high frequency responses are not limited to top 100 responses of training dataset.

The human evaluation results of experiments are shown in Table 5. We use agreement [Fleiss and others, 1971] to measure the inter-rater consistency. The agreement is represented by Fleiss’ kappa which ranges from 0 to 1. All the agreement of methods in Table 5 range from 0.4 to 0.6, which means all the results are in the same level of agreement that is “Moderate agreement” [Landis and Koch, 1977].

From Table 5, we find that two ensemble benchmarks have lower ratio of bad responses than Seq2Seq. However they perform worse than Seq2Seq on the ratio of good responses. The main reason is that those two methods generate more high frequency responses, which are semantically relevant to the posts but are not good.

Compared to benchmarks, AwE is superior to them. As for the comparison with Seq2Seq, our method generates more normal and good responses. This may result from two reasons. Firstly, AwE is an ensemble method, which leads to it generating more semantically relevant and grammatically correct responses. Secondly, the auto-evaluator in AwE helps our method to obtain more good responses. The auto-evaluator is trained by generated pairs and real pairs whose labels are different. It assigns $f(x_i) = -1$ if $y_i$ is high frequency and $y_i$ is not. Thus $(x_i, y_i)$ will obtain high weights in the next epoch, if $y_i$ is high frequency and $y_i$ is not. By this way, our method can get more information from low frequency and semantically relevant responses and reduce the probability of high frequency responses being generated, which leads to more good responses being generated. From Table 4, we can also see that our method generates less high frequency responses. In addition, two ensemble benchmarks do not have the auto-evaluator so that our method generates more good responses than them.

5.3 Case Study

Table 6 shows some responses generated by three benchmarks and AwE. From the first and the second examples, we can see that Seq2Seq generate semantically irrelevant responses to posts or responses with grammatical mistakes, whereas the three ensemble methods generate some semantically relevant responses without mistakes. This, as we conjecture, is caused by ensemble methods can always get better performance than single methods. From the third and fourth examples, we observe that AwE generate more satisfying responses than Maximum and Bagging. This is because, although the responses generated by Maximum and Bagging are semantically relevant to posts, they are high frequency. Instead, our method will decrease the weight of instances with high frequency responses in the training process so that our method generates less high frequency and more satisfying responses.

6 Conclusion

In this study, we propose Adaboost with Auto-Evaluation (called AwE) to improve performance of existing conversational models. Specifically, our method uses an auto-evaluator to evaluate the output generated by the conversational models so that the weights on the instances can be adjusted adaptively. We also theoretically analyze the upper bound of the training error for AwE when certain assumption on the auto-evaluators holds. From empirical experiments,
we conclude that AwE visibly outperforms existing conversational models and other ensemble methods.

Acknowledgments

Juncen Li, Fen Lin and Cheng Niu are supported by Tencent. Ping Luo and Ganbin Zhou were supported by the National Natural Science Foundation of China under Grant No. 61473274, and the National Key Research and Development Program of China under Grant No. 2017YFB1002104. We thank Wilson Tam and Yijun Xiao for their constructive advices. We also thank the anonymous IJCAI reviewers for their helpful feedback.

References


